DENOISING OF DEGRADED AUDIO SIGNALS USING WAVELET ANALYSIS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

EMRE BARIŞ AKSU

IN PARTIAL FULLFILMENT OF THE REQUIREMENTS FOR

THE DEGREE OF MASTER OF SCIENCE

IN

THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

DECEMBER 1998

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Tayfur Öztürk
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Fatih Canatan
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Supervisor

Asst. Prof. Dr Tolga Çiloğlu

Co-Supervisor

Prof. Dr. Mübeccel Demirekler

Examining Committee Members

Prof. Dr. Enis Çetin

Prof. Dr. Mübeccel Demirekler

Assoc. Prof. Kemal Leblebicioğlu

Asst. Prof. Dr. T. Engin Tuncer

Asst. Prof. Dr. Tolga Çiloğlu

ABSTRACT

DENOISING OF DEGRADED AUDIO SIGNALS USING WAVELET ANALYSIS

Aksu, Emre Barış
M.S., Department of Electrical and Electronics Engineering
Supervisor: Asst. Prof. Dr. Tolga Çiloğlu
Co-Supervisor: Prof. Dr. Mübeccel Demirekler
December 1998, 75 pages

The aim of this work is to present three methods that are based on wavelet analysis. The first method combines the well known denoising scheme, denoising by thresholding and wavelet packet analysis. The second method is developed during this study. It shapes the wavelet coefficients according to the energy distribution in the signal. The third method is a simplified version of denoising by entropy based best basis selection method that has been developed by J. Berger. Among the methods, method-3 has the best performance followed by method-2 and method-1.

Keywords: Audio Denoising, Wavelet Analysis, Wavelet Packets

ÖZ

DALGACIK İNCELEMESİ YÖNTEMİ İLE YIPRANMIŞ MÜZİK İŞARETLERİNİN GÜRÜLTÜDEN ARINDIRILMASI

Aksu, Emre Barış
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Yrd. Doç. Dr. Tolga Çiloğlu
Yardımcı Tez Yöneticisi: Prof. Dr. Mübeccel Demirekler
Aralık 1998, 75 sayfa

Bu çalışmada amaçlanan, dalgacık incelemesine dayalı olarak geliştirilmiş olan üç gürültü temizleme yönteminin sunulmasıdır. Birinci yöntem, dalgacık paketi analizi ve eşikleme ile işaretin gürültüden arındırılması yöntemlerine dayalıdır. İkinci yöntem, tez çalışması sırasında geliştirilmiş olup, dalgacık katsayılarının işaretteki enerji dağılımına göre şekillendirilmesi esasına dayanmaktadır. Üçüncü yöntem ise önceden J. Berger tarafından geliştirimiş olan entropi tabanlı en iyi temel bileşen seçimi ile işaretin gürültüden arındırılması yönteminin basitleştirilmiş halidir. Metodlardan en iyi performans metod-3'e aittir. Ardından sırayla metod-2 ve metod-1 gelmektedir.

Anahtar Kelimeler: Gürültü Temizleme, Dalgacık İncelemesi, Dalgacık Paketleri

To My Dear Wife Elif

ACKNOWLEDGMENTS

First, I would like to thank to my supervisors Asst. Prof. Dr. Tolga Çiloğlu and Prof. Dr. Mübeccel Demirekler for their support, suggestions, comments and guidance throughout my thesis, then to my defense committee members Prof. Dr. Enis Çetin, Assoc. Prof. Kemal Leblebicioğlu and Asst. Prof. Dr. T. Engin Tuncer.

Finally, special thanks to my family who encouraged and supported me throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

Denoising and reconstruction of audio signals is a recently developing area of interest. Mainly, the degradation in the audio signal depends on the media on which it is recorded as well as the degradation occurring in the course of time. Statistical methods are commonly used in denoising and reconstruction [19,20]. The success of these methods depends on the characteristics of the noise or the degradation on the audio signal. To illustrate, statistical methods based on Bayes rule have been used for denoising and reconstruction of the degradation caused by clicks , impulsive noise and the scratches on the recording media [21,22]. For the same kind of degradation, another method is based on the separation of the monophonic audio signal into two channels and applying statistical methods for denoising on them [23]. Another kind of distortion which should be considered is the deviation of the pitch periods of the instruments in the audio signal [24]. The main problems in denoising methods using statistical models are the complexity of the algorithms and the difficulties in the signal modelling phase.

The main idea behind another group of denoising and reconstruction algorithms is to make modifications in the frequency content of the signal. This group contains methods like simple low pass filtering or suppression of the frequency components (eg. DOLBY®). The main problem in this group of methods is the loss of high frequency content of the audio signal. Besides these, STFT (Short-Time Fourier Transform) is also used for the correction of the pitch variations of the degraded audio signal [25].

Different than the methods mentioned above, a relatively new method based on *wavelet analysis* is used for denoising and reconstruction of degraded audio signals [9,26,27]. Although the theoretical background on wavelets goes back to 1920s (used especially in quantum physics) , the applications on denoising audio signals have been mostly developed after 1994. Before that, some applications on denoising of statistical data were done [14]. But most of the results of these researches have been obtained after 1994.

Wavelet systems can be explained as *filter bank* structures with some interesting properties. By the help of these properties, they provide better localisation in both time and frequency in means of audio processing than the classical Fourier Transform or STFT tools. This is why they are very suitable for the analysis of signals like music where the musical scores can be thought as signals whose energies are concentrated at some frequency, and which can occur at any location in time, with a musically meaningful duration.

In this work, three denoising methods which use wavelet analysis techniques for denoising will be presented. These methods are concentrated on denoising of the noise type called the *background noise* which is also called the "hiss" sound present within the musical structure.

As for the organisation of the work, in Chapter 2, an introduction to *wavelets* and *wavelet expansion systems* will be given. The *multiresolutional* formulation of the wavelet systems will be discussed. The relationship between *filter banks* and the wavelet transform will be derived and explained. Also, the wavelet packet decomposition will be introduced.

In Chapter 3, the classical techniques on signal denoising via wavelet analysis will be given. The most well known method which is called '*denoising by thresholding*' will be discussed. Then, another method which is based on wavelet packet analysis will be explained.

In Chapter 4, the methods developed and the experimental work that has been done will be explained. Three methods will be discussed and explained. The experimental results will be given for each method.

In Chapter 5, the conclusions drawn from these results and suggestions for future work are presented.

In addition to these chapters, there are two appendices. In Appendix A, the necessary conditions and the properties of the scaling function and the wavelet are given. In Appendix B, the Matlab source codes of the three methods are given.

CHAPTER 2

THEORETICAL BACKGROUND ON WAVELETS

2.1 Introduction

In this chapter, a review of *wavelets* and *wavelet expansion systems* is presented. The *multiresolutional* formulation of the wavelet systems will be discussed. The relationship between *filter banks* and the wavelet transform will be derived and explained. Also, the wavelet packet decomposition will be introduced.

2.2 Wavelet Expansion Systems

Consider a real-valued function or signal $f(t)$. In order to have a better understanding of $f(t)$ for further analysis and processing, it can be written as a linear decomposition by

$$f(t)=\sum_{l} a_l \psi_l(t) \tag{2.1}$$

where l is an integer index for the sum (finite or infinite) , a_l are real – valued expansion coefficients, and $\psi_l(t)$'s are real valued functions of t which form the *expansion set*. If the representation of $f(t)$ in terms of

$\psi_l(t)$ is unique, then the set is called a *basis* for its span. If the following condition is satisfied

$$\langle \psi_k(t), \psi_l(t) \rangle = \int \psi_k(t) \psi_l(t) dt = 0 \qquad k \neq l, \tag{2.2}$$

then the basis is called an orthogonal basis and the coefficients a_k can be calculated by the inner product

$$a_k = \frac{\langle f(t), \psi_k(t) \rangle}{\langle \psi_k(t), \psi_k(t) \rangle} = \frac{\int f(t) \psi_k(t) dt}{\int \psi_k(t)^2 dt} \tag{2.3}$$

By simply substituting (2.1) into (2.3) and using (2.2) , one can prove this result.

Table 2. 1. Examples of orthogonal and non-orthogonal basis functions

Name	*Basis functions* ($\psi_k(t)$)	*Type*
Fourier Series	$\sin(k\omega_0 t), \cos(k\omega_0 t)$	Orthogonal
Taylor Series	t^k	Non-orthogonal

For the *wavelet expansion*, a two parameter system is used. For this system (2.1) becomes

$$f(t) = \sum_k \sum_j a_{j,k} \psi_{j,k}(t) \tag{2.4}$$

where both j and k are integer indices and $\psi_{j,k}(t)$ are called the *wavelet expansion functions* which usually form an orthogonal basis. In (2.4) , $a_{j,k}$ are called *The Discrete Wavelet Transform* (DWT) of function or signal *f(t)*. So, (2.4) is called the *Inverse Wavelet Transform* (IWT).

2.2.1 General Characteristics of Wavelet Systems

There are many different wavelet systems but all satisfy six general characteristics [1][2][3]:

1. A *wavelet system* is a two-dimensional expansion set for some class of one or higher dimensional signals.
2. The *wavelet expansion* gives a time-frequency *localization* of the signal. This means most of the energy of the signal is well represented by a few $a_{j,k}$ coefficients. This localization is in both time and frequency simultaneously. According to this property, one can say that, a wavelet system is much like a *musical score* where the location of the notes tells when the tones occur and what their frequencies are.
3. The calculation of the coefficients from the signal can be done *efficiently*. On many wavelet expansion systems, the coefficients can be calculated with *O(N)* operations. The meaning of this is that, the number of floating-point multiplications and additions increase *linearly* with the length of the signal. Most of the transformations like the Fast Fourier Transform (FFT) require *O(Nlog(N))* operations.
4. All so-called first-generation wavelet systems are generated from a single scaling function or wavelet by simple *scaling* and *translation*. The basis functions $\psi_{j,k}(t)$ are generated from the function $\psi(t)$ (sometimes called the *generating* or the *mother* wavelet) by

$$\psi_{j,k}(t) = 2^{j/2}\psi\left(2^{j}t - k\right) \qquad j,k \in \mathbf{Z} \tag{2.5}$$

where parameter *j* represents the frequency (or logarithm of scale) and *k* represents the time or space location. Note that the $2^{j/2}$ factor maintains a constant norm independent of the scale *j*.

5. Most of the wavelet systems (including the ones used in this thesis work) satisfy the *multiresolution* conditions. What is meant from this can be explained with the following example. If a set of signals can be represented by a weighted sum of $\psi(t-k)$, then a larger set (also including the one previously mentioned) can be represented by a weighted sum of $\psi(2t-k)$.

6. The *lower resolution* coefficients can be calculated from the higher resolution coefficients by a tree-structured algorithm called a *filter bank*. This method offers a very efficient way of calculating the expansion coefficients and also relates the subject of wavelet transformation to the area of d*igital signal processing*.

2.2.2 Examples of Wavelets

In order to give the reader some insight on how the wavelets look like, some well known *mother wavelets* and their shifted and scaled versions are given in the following figures.

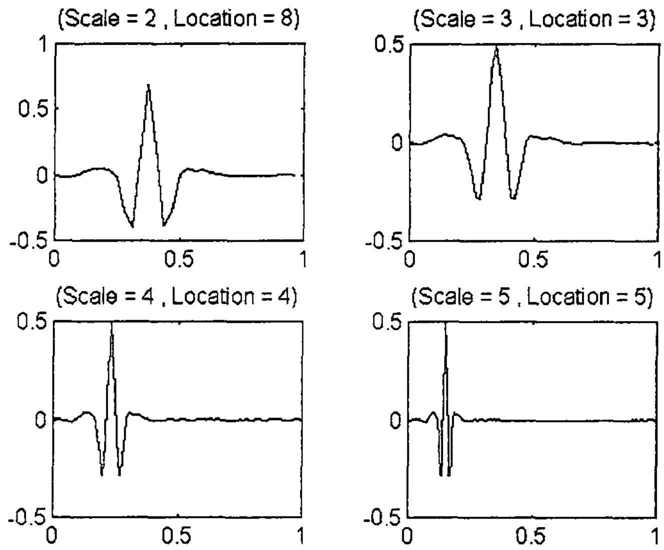


Figure 2. 1. Coiflet – 3 mother wavelet and its scaled and shifted versions

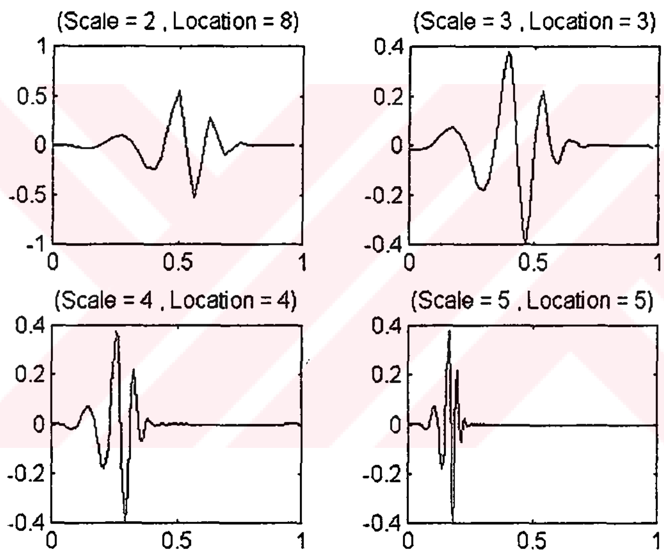


Figure 2. 2. Daubechies –12 mother wavelet and its scaled and shifted versions

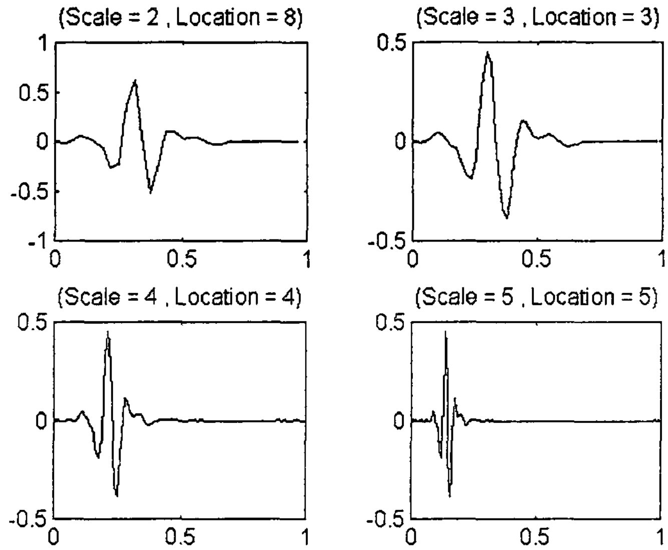


Figure 2. 3. Symmlet-7 mother wavelet and its scaled and shifted versions

2.3 Multiresolutional Formulation Of Wavelet Systems

Before going into the details of the multiresolutional formulation, one has to mention the role of the so called *scaling function*. The scaling functions can best be explained by signal spaces convention.

Just like the wavelet functions, the scaling functions are also shifted and scaled versions of a single scaling function, which is called the *mother scaling function*. The representation is as follows:

$$\varphi_k(t) = \varphi(t-k) \qquad k \in \mathbf{Z} \tag{2.6}$$

and the signal space spanned by this function set can be represented as

$$v_0 = \overline{\underset{k}{Span}\{\varphi_k(t)\}} \tag{2.7}$$

where k is an integer in $(-\infty,+\infty)$ and the over bar represents closure. So, one can admit the following representation:

$$f(t)=\sum_k a_k\varphi_k(t) \qquad \exists\ f(t) \in v_0. \tag{2.8}$$

The size of signal space spanned can be increased by changing the time scale of the scaling function. A two-dimensional family of scaling functions can be generated as follows:

$$\varphi_{j,k}(t)=2^{j/2}\varphi(2^j t-k) \tag{2.9}$$

whose span over k is

$$v_j=\overline{\underset{k}{Span}\{\varphi_k(2^j t)\}}=\overline{\underset{k}{Span}\{\varphi_{j,k}(t)\}} \tag{2.10}$$

this means that if $f(t)$ is in v_j then it can be expressed as

$$f(t)=\sum_k a_k\varphi(2^j t+k) \tag{2.11}$$

It can be intuitively seen that for j>0 , the span can be larger since the scaling function $\varphi_{j,k}(t)$ is narrower and it is translated in smaller steps. Therefore, as j increases , the scaling function can represent finer detail.

The nested signal spaces spanned by the scaling functions can be shown as follows [4]:

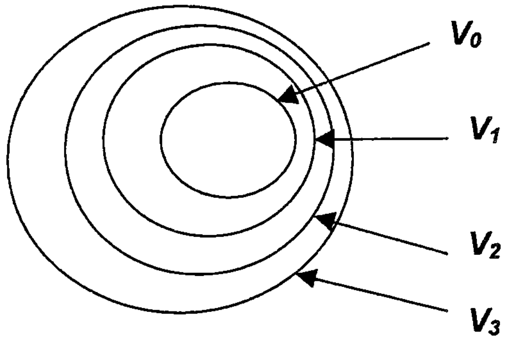


Figure 2. 4. The nested signal spaces

Note that the scaling function of one higher resolution also spans the space which is defined by the one lower resolution's scaling function. This means that $\varphi(t)$ can be expressed in terms of a weighted sum of shifted versions of $\varphi(2t)$, or

$$\varphi(t) = \sum_{n} h(n)\sqrt{2}\varphi(2t - n) \qquad (2.12)$$

Equation (2.12) is called *the refinement* or *the multiresolution analysis equation*. In this equation, h(n) is a sequence of real or complex numbers. There are of course some necessary conditions that must be satisfied by $\varphi(t)$ for the equation to posses a solution. These necessary conditions and also the properties of the scaling function can be found in Appendix A.

Most of the time it is advantageous and efficient to deal with the signals present in the *difference* signal spaces. This is exactly what the wavelets do.

What is meant can be expressed by the following graph. Let $\mathbf{W}_i$'s represent the *difference spaces* between the signal spaces of different resolutions ($\mathbf{V}_i$'s).

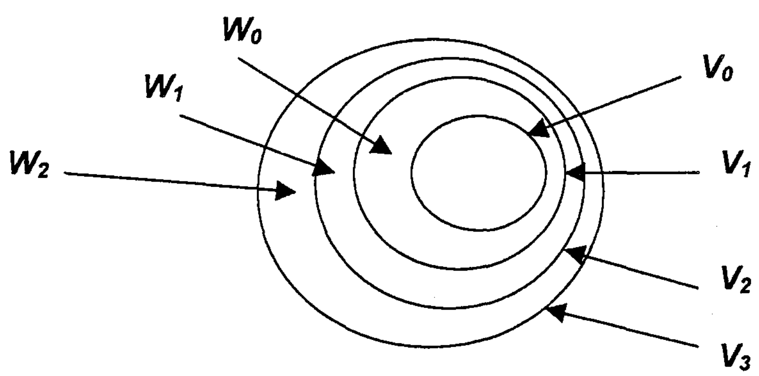


Figure 2. 5. The difference signal spaces

In this figure, each difference space is spanned by the wavelet functions. Since each wavelet function resides in the space spanned by the next higher resolution scaling function, it can be represented by a weighted sum of the scaling function which spans the one higher resolution. So , one can relate the wavelet function to the scaling function of one higher resolution as follows:

$$\psi(t) = \sum_{n} h_1(n)\sqrt{2}\varphi(2t-n) \qquad n \in \mathbf{Z} \qquad (2.13)$$

for some set of coefficients $h_1(n)$. The wavelets span the difference or the orthogonal complement spaces. The wavelet coefficients $h_1(n)$ are related to the scaling function coefficients as follows:

$$h_1(n) = (-1)^n h(N-1-n) \qquad (2.14)$$

where N is the (even) length of h(n). A brief derivation of (2.14) can be found in [1]. The basic wavelet properties can be found in Appendix A.

The wavelets at each scale and time shift (j,k) can be generated from the mother wavelet as follows:

$$\psi_{j,k}(t) = 2^{j/2}\psi(2^j t - k) \qquad (2.15)$$

Now, one can represent a signal $f(t)$ in $\mathbf{L}^2(\mathbf{R})$ by a series expansion of both the wavelet functions and the scaling functions.

$$f(t) = \sum_{k} c(k)\varphi_k(t) + \sum_{j=0}^{\infty}\sum_{k} d(j,k)\psi_{j,k}(t) \qquad (2.16)$$

Equation(2.16) can be written in an open form as follows:

$$f(t)=\sum_k c_{j_0}(k)2^{j_0/2}\varphi\left(2^{j_0}t-k\right)+\sum_{j=j_0}^{\infty}\sum_k d_j(k)2^{j/2}\psi\left(2^j t-k\right) \tag{2.17}$$

where j_o indicates the first (or the lowest) level of resolution. Note that j_o can be any integer value, but it must be noted that, in discrete signals, where the length of the signal is finite, the highest resolution level is bounded by the length of the signal.

The coefficients c_j (scaling coefficients) and d_j (wavelet coefficients) are iterated from one higher resolution coefficients via the following two equations:

$$c_j(k)=\sum_m h(m-2k)\ c_{j+1}(m) \tag{2.18}$$

$$d_j(k)=\sum_m h_1(m-2k)\ c_{j+1}(m) \tag{2.19}$$

It must be noted that, if the scaling function is well-behaved, then at a relatively high scale, it looks like a Dirac delta function. This property tells us that, at a relatively high scale, the expansion coefficients (the scaling coefficients) are very close to the signal's samples. This property gives us the initial coefficients (*c_{j+1}'s)* necessary for the iterative relations defined in equations (2.18) and (2.19).

2.4 An Example On Multiresolutional Decomposition

In the following figures, the projection of the Doppler signal onto signal spaces V_i and W_i are given. Figure 2.6 shows the decomposition of the Doppler signal to 10 signal spaces using Daubechies-8 as the mother wavelet function. Figure 2.7 shows the decomposition of the Doppler signal to 9 *difference* signal spaces using Daubechies-8 as the mother wavelet function.

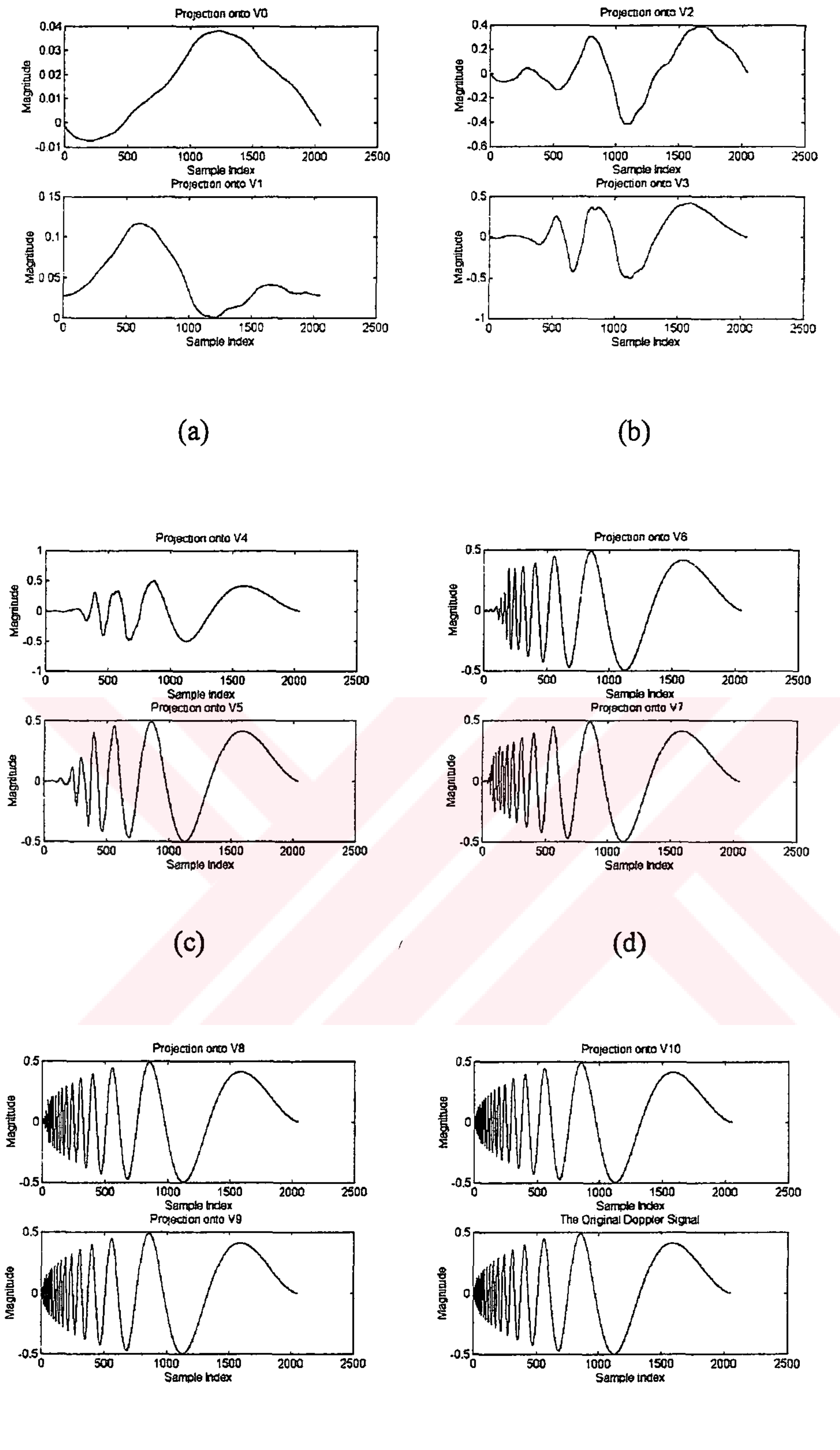
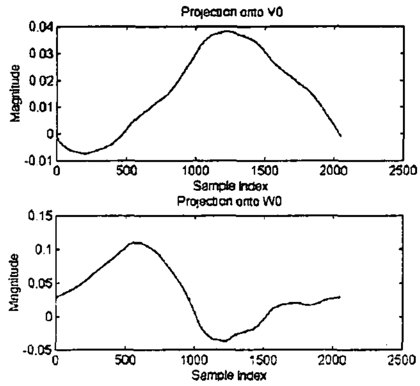
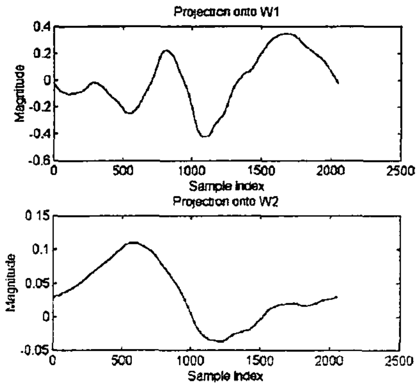
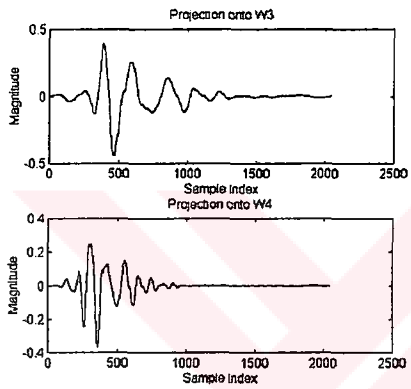


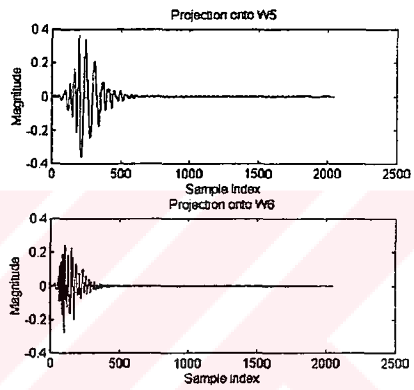
Figure 2. 6. The projections of the Doppler signal onto signal spaces V_I



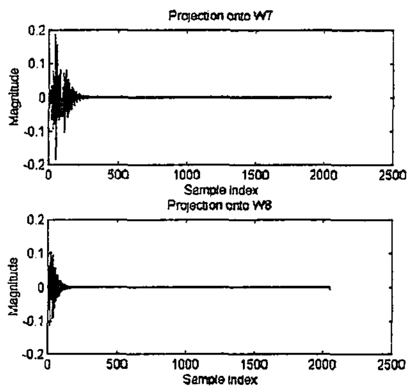


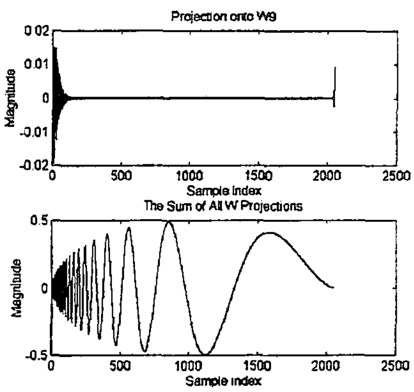
(a) (b)





(c) (d)





(e) (f)

Figure 2. 7. The projections of the Doppler signal onto signal spaces W_i

2.5 Representation of Wavelet Transform Using Filter Banks

In equations (2.18) and (2.19), it is shown that one lower resolution's scaling and wavelet function coefficients can be found from the one higher level's scaling coefficients. These equations can be realized via filtering and down-sampling conventions which are very much used in digital signal processing area. The realisation is as follows:

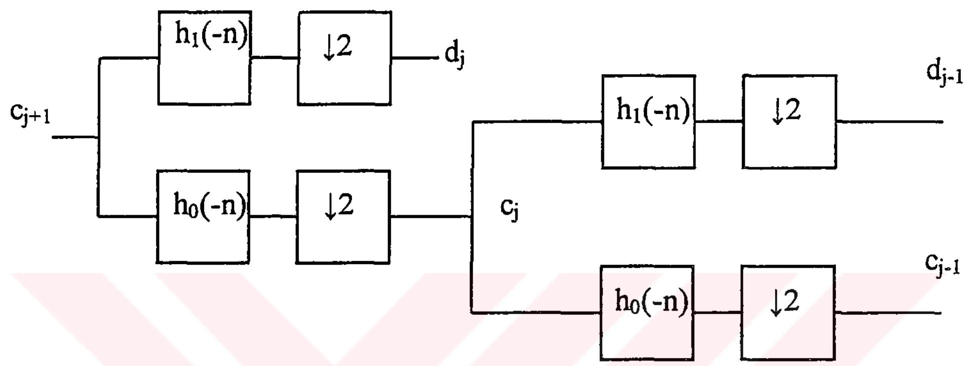


Figure 2. 8. Filter bank structure of obtaining one lower resolution's scaling and wavelet coefficients from one higher resolution's scaling coefficients

In the above realisation, the symbol '↓' stands for *down-sampling* or *decimation* and $h_0(-n)$ is the same as $h(-n)$. The higher level scaling coefficients c_{j+1} are used to extract the one lower level scaling coefficients (c_j) and wavelet coefficients (d_j). the same filtering and decimation structure is used for the scaling coefficients cj and so on. In fact, the filter implemented by $h_0(-n)$ is a low pass filter , and the one implemented by $h_1(-n)$ is a high pass filter. It must be noted that the *average* number of data points out of this system is the same as the number in. As the number is doubled by having two filters, it is halved by the decimation back to the original number. What is meant here is that, there is a possibility of no loss in the information beared by c_{j+1}. This possibility becomes a reality when the filter type called *Quadrature Mirror Filters* with perfect reconstruction capability is used for places of $h_0(-n)$ and $h_1(-n)$. More information on quadrature mirror filters can be found in [5].

Now let us look at the system from the frequency domain point of view. Let us define the signal spaces represented by the scaling coefficients as V_j and the signal spaces represented by the wavelet coefficients (the *difference* signal spaces) as W_j. Note that the index '*j*' decreases with the resolution in this representation. The following figure represents the filter realisation's effect on the signal spaces:

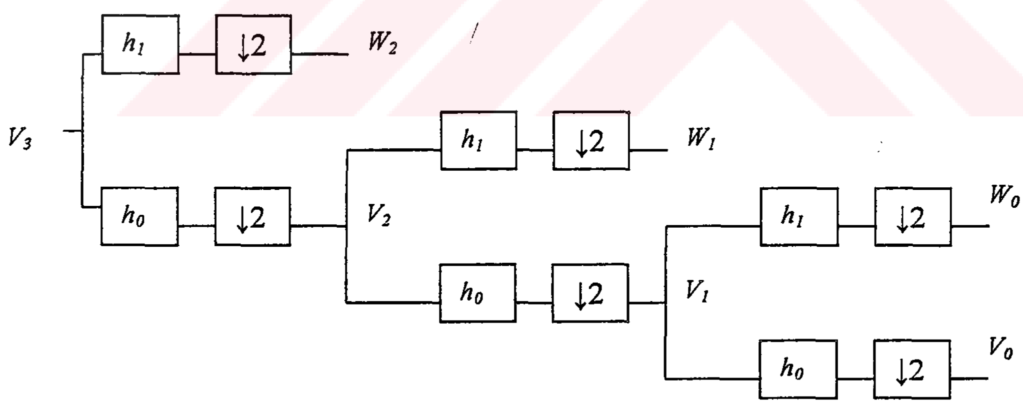


Figure 2. 9. The filter bank realisation's effect on the signal spaces

As h_0 and h_1 are low pass and high pass filters respectively, the frequency domain is split as represented in the following figure:

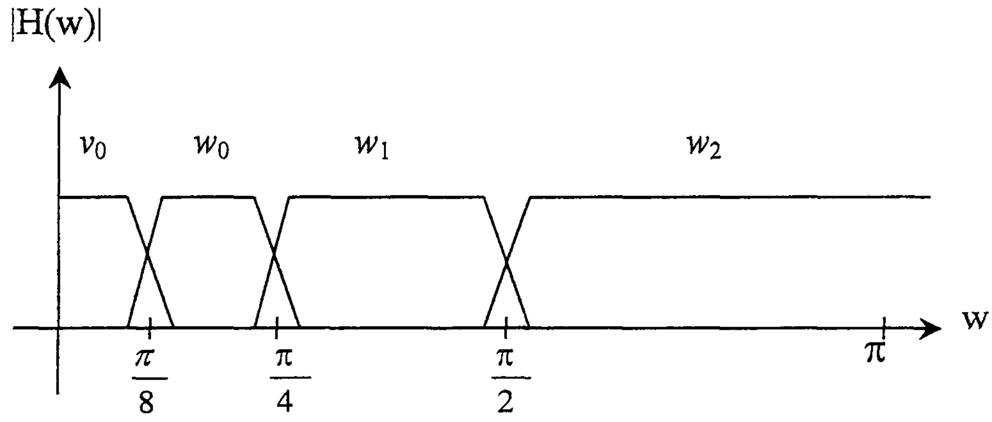


Figure 2. 10. The frequency domain partitioning obtained by the filter bank realisation

As can be seen from the above figure, the classical wavelet system results in a logarithmic frequency resolution, which is very useful in speech and audio applications. The reason is that, the human ear is sensitive to frequencies in a logarithmic scale (more sensitive (better tracking of changes in frequency) to low frequencies and less sensitive (worse tracking of changes in frequency) to high frequencies.

For the synthesis part, the following equation is obtained [1]:

$$c_{j+1}(k) = \sum_m c_j(m) h(k-2m) + \sum_m d_j(m) h_1(k-2m) \tag{2.20}$$

This equation can also be realized as a two-band synthesis bank as follows:

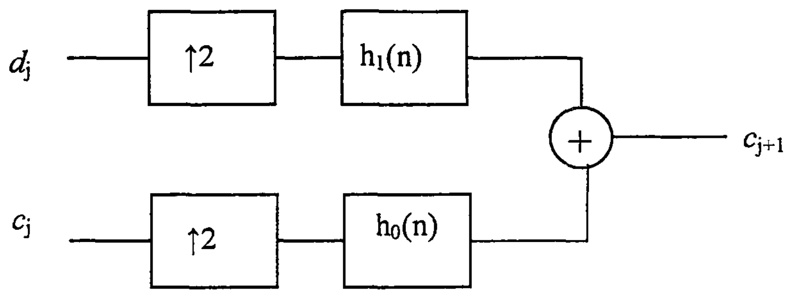


Figure 2. 11. The two-band synthesis bank

By using the filter bank structure above in an iterative manner , one can reach the scaling coefficients at the highest possible resolution, which is the original signal itself.

2.6 Wavelet Packets and Wavelet Packet Decomposition

In classical wavelet transform, it can be seen that only the low-pass part of the filter bank structure is iterated to get the difference spaces. This only allows a resolutional analysis at low frequency contents. One can also have a resolutional analysis at each (or selected) *difference signal space* spanned by wavelet functions via *wavelet packet analysis*.

Wavelet packets are functions that are used to separate a wavelet function's span into subspaces. This can be illustrated by the following figure :

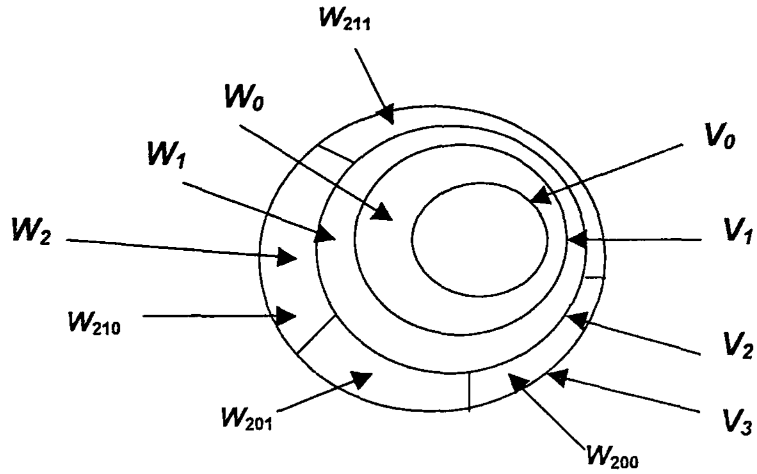


Figure 2. 12. Separation of the wavelet function's span by the wavelet packets

From the filter bank point of view, the separation can be realized by iterating the high pass part of the filter bank structure. The following figure illustrates the full binary tree for the three-scale wavelet packet transform.

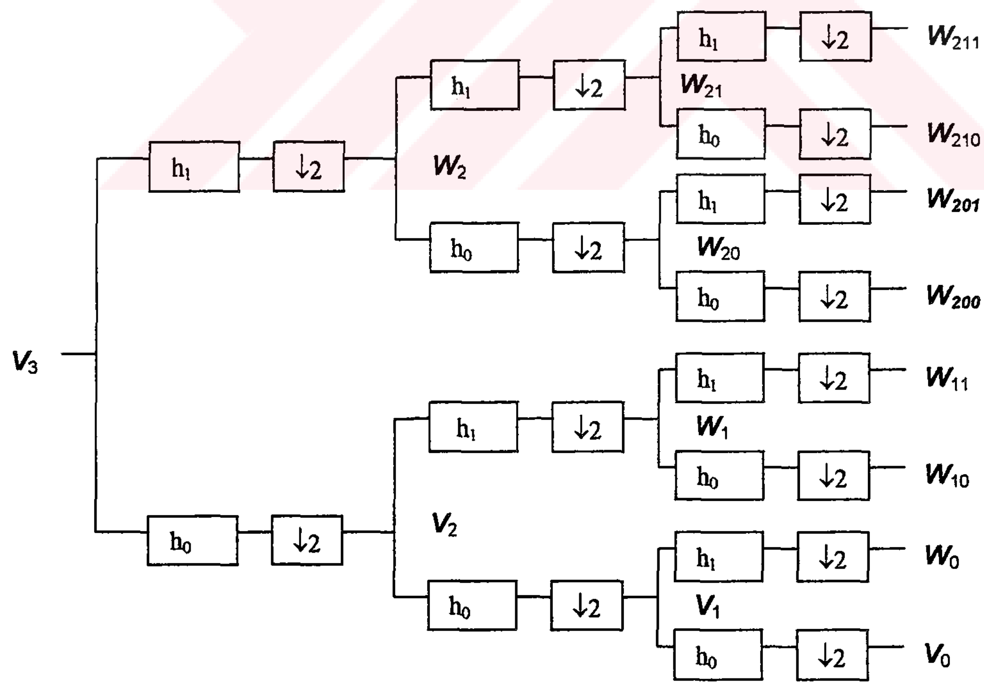


Figure 2. 13. The filter bank structure of wavelet packet decomposition

If one looks from the frequency domain point of view, the following frequency partition is obtained:

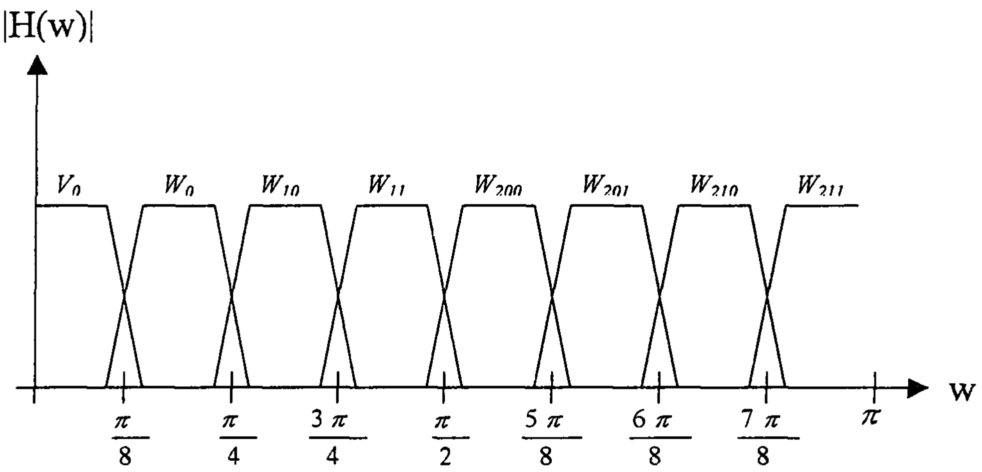


Figure 2. 14. The frequency domain partitioning obtained by the wavelet packet decomposition

It can be seen from the frequency domain picture of the wavelet packet analysis that, wavelet packets allow signal analysis at finer resolution.

CHAPTER 3

THE WAVELET DENOISING TECHNIQUES

3.1 Introduction

In this chapter, the classical techniques on signal denoising via wavelet analysis will be given. The most well known method which is called '*denoising by thresholding*' will be explained. Then , another method which is based on wavelet packet analysis will be reviewed.

3.2 Denoising By Thresholding

Most of the study on the wavelet denoising is done by Donoho [7,8,14], Johnstone [15,16,17] and Coifman[13]. The method of denoising is based on taking the discrete wavelet transform of a signal, passing this transform through a threshold, which removes the coefficients below a certain value, then taking the inverse wavelet transform. This method has the ability to remove the noise because of the *concentrating* ability of the wavelet transform. If the energy of a signal is concentrated in a small number of wavelet dimensions, its coefficients will be relatively large compared to any other signal or noise that has its energy spread over a large number of coefficients. This means that thresholding the wavelet transform will remove the low amplitude noise or the undesired signal in the wavelet domain, and an inverse transform will then retrieve the desired signal with little loss of detail. It must also be noted that, if the energy of the signal being analysed is very low, then most of the coefficients remain below the threshold although they support a meaningful signal. Then the operation has no difference than a classical filtering operation. To overcome this problem, the threshold must be selected properly.

In classical Fourier–based signal processing, the system is arranged such that the signal and noise overlap as little as possible in the frequency domain and linear time-invariant filtering will approximately separate them. But when their Fourier spectra overlap, they can not be separated. Using linear wavelet or other time–frequency or time-scale methods, one can try to choose basis systems such that in that coordinate system, the noise and the signal overlap as little as possible, and the separation is possible.

The nonlinear method (denoising by thresholding) is entirely different. The spectra of the noise and the signal can overlap as much as they want. The idea is to have the amplitude, rather than the location of the spectra be as different as possible. This allows clipping, thresholding, and shrinking of the amplitude of the transform to separate signal from the noise. It is the localizing or the concentrating properties of the wavelet transform that makes it particularly effective when used with these nonlinear methods.

The basic ideas of thresholding the wavelet transform is developed by Donoho [7,8]. Assume a finite length signal in R^n with additive noise of the form

$$y_i = x_i + \varepsilon n_i \qquad i=1,\dots,N \tag{3.1}$$

as a finite length signal of observations of the signal x_i that is corrupted by i.i.d. zero mean, white Gaussian noise $\boldsymbol{n}$ ($\boldsymbol{n}$ = $[n_1\ n_2 \dots n_N]$) with standard deviation 1 . The goal is to recover the signal $\boldsymbol{x}$ ($\boldsymbol{x}$ = $[x_1\ x_2 \dots x_N]$) from the noisy observations $\boldsymbol{y}$ ($\boldsymbol{y}$ = $[y_1\ y_2 \dots y_N]$). Let W be a left invertible wavelet transformation matrix of the discrete wavelet transform. Then equation (3.1) can be written in the transformation domain as follows:

$$\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{N} \tag{3.2}$$

where capital letters denote the variables in the transform domain, i.e.

$$\boldsymbol{Y} = \mathrm{W}\boldsymbol{y} \quad , \quad \boldsymbol{N} = \mathrm{W}\boldsymbol{n} \tag{3.3}$$

In the following equations and results, Donoho's approach are taken as a reference. What is meant from this statement is that, the wavelet transform is considered to be an *orthogonal* transform and W matrix is square, with the property $W^{-1}=W^T$.

Let $\hat{\boldsymbol{X}}$ denote an estimate of $\boldsymbol{X}$, based on the observation $\boldsymbol{Y}$. We consider diagonal linear projections

$$D = \mathrm{diag}(\delta_1,\dots,\delta_N), \quad \delta_i \in \{0,1\} \qquad i=1,\dots,N \tag{3.4}$$

which give rise to the estimate

$$\hat{\boldsymbol{x}} = \mathrm{W}^{-1}\hat{\boldsymbol{X}} = \mathrm{W}^{-1}\mathrm{D}\boldsymbol{Y} = \mathrm{W}^{-1}\mathrm{DW}\boldsymbol{y} \tag{3.5}$$

The estimate $\hat{X}$ is obtained by simply keeping or zeroing the individual wavelet coefficients. Since the important thing is the l_2 error, the risk measure is defined as

$$R(\hat{X},X)=E\left[\|\hat{x}-x\|_2^2\right]=E\left[\left\|W^{-1}(\hat{X}-X)\right\|_2^2\right]=E\left[\|\hat{X}-X\|_2^2\right] \tag{3.6}$$

The optimal [7] coefficients in the diagonal projection scheme are $\delta_i = 1$ if $X_i > \varepsilon$ which means that only those values of Y where the corresponding elements of X are larger than ε are decreased by an amount of ε and all others are set to zero (the main idea of *soft thresholding*). This leads to the ideal risk

$$R_{id}(\hat{X},X)=\sum_{n=1}^{N}\min\left(|X_i|^2,\varepsilon^2\right) \tag{3.7}$$

Equation (3.7) implies that, the ideal risk is the summation of the minimum of the magnitude square of X_i and the threshold value ε.

In practice, the ideal risk can not be attained because we do not know the vector x, and so X. However equation (3.7) is important, it gives a lower limit for the l_2 error.

In summary, Donoho's method for denoising can be presented as follows:

1. Compute the DWT ie., $Y=Wy$.
2. Perform the thresholding in the wavelet domain, according to so-called ***hard thresholding :***

$$\hat{X}_i = T_h(Y_i,\varepsilon)=\begin{cases} Y_i, & |Y_i| \ge \varepsilon \\ 0, & |Y_i| < \varepsilon \end{cases} \qquad i=1,2\ldots,N \tag{3.8}$$

or

soft thresholding :

$$\hat{X}_i = \mathrm{T}_s(Y_i, \varepsilon) = \begin{cases} \operatorname{sgn}(Y_i)(|Y_i| - \varepsilon), |Y_i| \geq \varepsilon \\ 0, |Y_i| < \varepsilon \end{cases} \qquad i=1,2,\ldots,N \qquad (3.9)$$

3. Compute the inverse DWT , ie. $\hat{x} = \mathrm{W}^{-1}\hat{X}$

This simple scheme has several properties. If one employs soft thresholding, then the estimate is with high probability at least as *smooth* as the original signal. Hard thresholding gives better results in terms of the l_2 error, but the denoised signal is *not* guaranteed to be smooth. It may have abrupt discontinuities.

It must be noted that, although denoising by thresholding works well on *"statistical signals"* like sonar data, ECG signals ,etc..., the same thing can not be said for musical and audio signals. Using this method *alone* on musical signals cause loss of musical content at an unbearable level. While listening to the so called denoised musical signal, one can hear another kind of distortion added. The cause is mostly the threshold level. This level is fixed by the help of the signal segment that contains no meaningful data (in musical signals , this corresponds to the silence period on the recording which appears before the music). The threshold is mostly selected as the *standard deviation* of the signal at this segment [7]. Experimental results show that , this selection is not too realistic for a musical signal. The main reason is that, this level is fixed for the whole signal, which means that if a noise burst occurs at any location, the threshold level may not be high enough to remove the noise, and if it were that high, it may cause loss of musical content at that location, which causes another kind of distortion added to the signal (which is mostly the case for a musical signal). So, working with thresholding alone is not sufficient to achieve denoising. A high resolution analysis is needed at this situation, which is used in the following method .

3.3 Denoising By Using Wavelet Packets

Denoising of musical signals using wavelet packet analysis is mostly based on the selection of *best basis* among several orthonormal *basis functions* , which correspond to wavelet packets at different time-frequency resolutions [9]. The idea of *best basis selection* will be described next.

Let us define the following sequence of functions:

$$W_{2n}(x)=\sqrt{2}\sum_{k} h_0(k)\, W_n(2x-k) \tag{3.10}$$

$$W_{2n+1}(x)=\sqrt{2}\sum_{k} h_1(k)\, W_n(2x-k) \tag{3.11}$$

The function $W_0(x)$ can be identified with $\varphi(x)$ and $W_1(x)$ with $\psi(x)$ which were defined in Chapter-2 by the equations (2.12) and (2.13) respectively. Also, the filters $h_0(k)$ and $h_1(k)$ are called *quadrature mirror filters* with perfect reconstruction capability and the relation between them were given in Chapter-2 by the equation (2.14).

All possible basis sets are extracted from the signal under consideration by building a *basis tree*. This tree consists of the wavelet packet coefficients obtained by applying the binary tree structure given in Figure 2.13 in Chapter-2. The tree is expanded to a desired level. In order to choose the best basis, a cost function must be selected and the basis with the lowest cost for the signal must be chosen.

The choice of the cost is arbitrary. Music should have more structure than the corrupting noise, and in a basis well suited to the signal, the coefficients corresponding to the musical content should have relatively large amplitudes and be few in number. This requirement is precisely what is measured by a *Low Shannon Entropy* [9]. So, this entropy value is used as a cost function.

The cost function is associated to the expansion of the signal in some particular orthogonal basis $\{\mathbf{w}_i\}$, with the corresponding coefficients as $\mathbf{y}_i$. In this notation, the Shannon entropy is defined as follows:

$$-\sum_{j=1}^{\text{length of } y_i} y_{ij}^2 \log_2 y_{ij}^2 \tag{3.12}$$

Note that $\mathbf{y}$ must be normalised with respect to its magnitude. The Shannon entropy measures how well $\mathbf{y}$ is represented by a particular basis set. There are two extreme cases which must be taken into consideration in order to have an insight about the operation. The first extreme case is when all y_{ij} values are 0 except one of them. Then, the entropy is 0; the subset of the basis set which contains a single vector is the best possible representation for the signal. The second extreme case is when all y_{ij} values have equal magnitude. The meaning of this is $\mathbf{y}$ is completely spread out relative to its basis. The entropy in this case is $\log_2 n$, where n is the number of entries in the vector. The closer the entropy to 0, the better the basis representing the signal, which means that there are only a few relatively large y_{ij} values, the rest is small. If the entropy is close to $\log_2 n$, all ,or most of the coefficients of $\mathbf{y}_i$ are of roughly equal in size.

Now let us consider the basis tree structure and the selection of the best basis. Figure 3.1 shows a basis tree structure up to the 4th resolution.

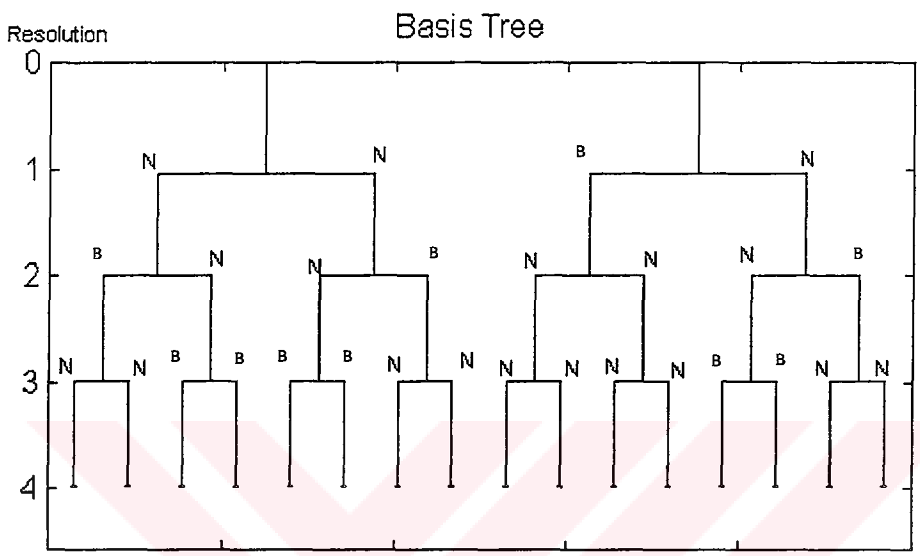


Figure 3. 1. Basis tree structure up to the 4th resolution

(letter 'N' stands for node and 'B' stands for best basis node)

For each node (representing a basis vector or can also be called the filter coefficients at a certain resolution) which is marked as "N" or "B" in Figure 3.1, we have an entropy value associated to it. Each node in the basis tree has two children nodes, which are the one higher resolution basis. The process of selecting the best basis is done in the following way. For each (parent) node, the entropy is compared to the sum of the entropies of the children nodes it has. If the parent's cost is less, it is marked to be kept, and no nodes below it are kept; otherwise the parent is marked not to be kept.

This process provides a representation for the signal being considered that has the lowest cost for that tree. In Figure 3.1, the nodes that are marked as kept are represented by the letter "B".

After the *best* basis is selected, a *coherent* and a *noisy* signal must be extracted. Let the input signal be **y**, with $\mathbf{y}_i$ the coefficients for the basis $\{\mathbf{w}_i\}$. The coefficients $\mathbf{y}_i$ are ordered by decreasing magnitude, that is,

$$\|y_1\| \geq \|y_2\| \geq \ldots \tag{3.13}$$

Now, another entropy which is called *normalized entropy* is defined as follows[9]:

$$E_i = \frac{2^{\text{entropy}_i}}{\text{vector size of } \mathbf{y}_i} \tag{3.14}$$

It is easy to see that $0 < E_i \leq 1$ with E_i close to 0 if the entropy is small. Then, the vectors of successive tails $\mathbf{V}_1$, $\mathbf{V}_2$,... where $\mathbf{V}_1$=**y**, $\mathbf{V}_2$ being **y** minus the signal represented by $\mathbf{w}_1$, $\mathbf{V}_3$ being $\mathbf{V}_2$ minus the signal represented by $\mathbf{w}_2$ and $\mathbf{V}_j$ being $\mathbf{V}_{j-1}$ minus the signal represented by $\mathbf{w}_{j-1}$ are considered. The signal represented by $\mathbf{w}_j$ is practically found by inverse wavelet packet transforming with only considering the coefficients $\mathbf{y}_j$ as input and taking all other coefficients (which represent some other resolution) as zero.

For each $\mathbf{V}_j$, starting at j=1, normalized entropy E_j is calculated. As soon as E_j is greater than or equal to some specified threshold α, $0< \alpha <1$, the process stops. If the process is stopped at, for example, $j=j_0$, then the signal $\mathbf{V}_1$ minus the signal $\mathbf{V}_{j0}$ is called the *coherent* portion of the signal, and the rest is called noise.

The threshold α is specified in advance. For audio signals, 0.30 or 0.35 is a good threshold to try , which means choosing 3 and 10% of the coefficients for the coherent part [9].

CHAPTER 4

DENOISING OF DEGRADED AUDIO SIGNALS VIA WAVELET ANALYSIS

THE METHODS USED AND THE EXPERIMENTAL WORK

4.1 Introduction

In this chapter, the methods developed and the experimental work that has been done will be explained. Three methods will be discussed. The experimental results will be given for each method. The input signal used for testing the methods is an old recording of classical Turkish music singer Zeki Müren.

4.2 Method-1

In Method-1, both the thresholding and the wavelet packet analysis tools are used [10]. In addition to these, some decisions are made according to the relative energy contents of the sub-signals obtained in the process. The process uses an *overlapping windowed* version of the original signal. Hanning window is used with an overlap of half the window length. Various windows of lengths 1024, 2048 and 4096 samples have been used.

The method can be explained as the follows:

1. Decompose the *windowed* signal into sub-signals using wavelet

transform.

This process corresponds to the *wavelet transformer* and the *Sub-Signal Generator* blocks in Figure 4.1. In this process 'Daubechies' or 'Symmlet' are used as mother wavelet functions[1]. For each resolution of decomposition, a sub-signal is generated from the wavelet coefficients corresponding to that resolution. In exact, signals in each *difference signal space* which have been mentioned in Chapter-2 are found.

2. Select the sub-signals which have relatively high energy and reject the others.

   This process corresponds to *Sub-Signal Selector 1* block in Figure 4.1. During this process, the energies of the sub-signals are found. By energy , we mean the summation of the squares of the samples of the sub-signal. If the ratio of the energy of sub-signal to the energy of the original signal is smaller than a threshold (0.05 which is a normalised value is used in the experiments) it is rejected (deemed noise signal) , otherwise it is kept.

3. Select the sub-signals whose energies are below a threshold value and apply wavelet packet transform to them to obtain better resolution in frequency.

   This process corresponds to the *Sub-Signal Selector 2* and *Wavelet Packet Transformer* blocks in Figure 4.1. This process is included for analysis of sub-signals under suspect (containing relatively high noise components) at a better resolution. The threshold level mentioned here highly depends on the noise level of the signal. It is selected as 0.1 (which is normalised to 1) in the experiments. Also Symmlet mother wavelet is used for the packet decomposition.

4. Select the packet coefficients deemed coherent (containing audio information) and reconstruct the sub-signal.

This process corresponds to *the Audio Content Bearing Packets Selector* and *Inverse Wavelet Packet Transformer* blocks in Figure 4.1. This step looks like the second step. There are two main methods for selecting the coherent part's wavelet packet coefficients. The first one is applying soft thresholding on the packet coefficients as explained in Chapter-3. After the thresholding process, the sub-signal is reconstructed back. The second method depends on rejection of a *group* of wavelet packets completely depending on their energy level, ie., hard thresholding. This method is easier than the first one and the threshold level is chosen between 0.3-0.4 (normalised) in the experiments.

5. Obtain the denoised signal by summing the sub-signals selected from Step 2 and reconstructed after Step 4.

This process corresponds to the S*ub-Signal Adder* block in Figure 4.1. In this last stage, the sum of the sub-signals gives the denoised signal. The flow chart of the Method is as follows:

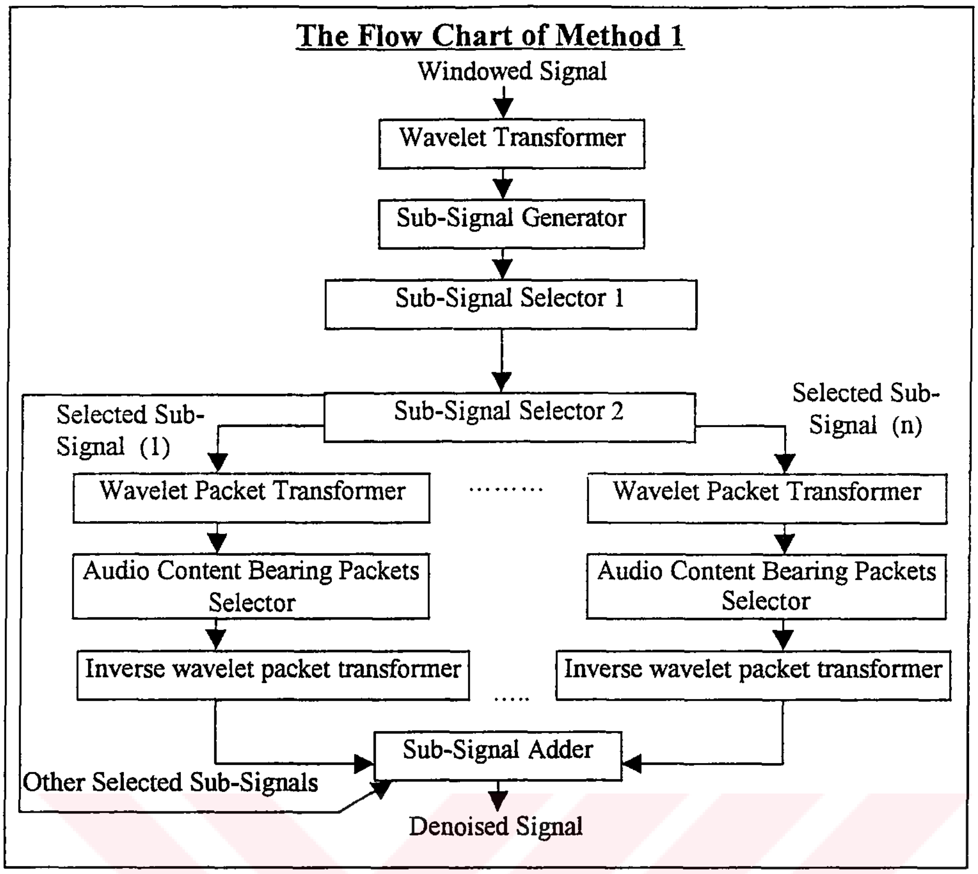


Figure 4. 1. The flow chart of Method-1

4.3 Method-2

The second method which will be introduced is based on the shaping of wavelet coefficients by the help of the energies of the sub-signals [10]. This method is based on the fact that, the energy of the noise on the whole is lower than the energy of the musical content in a significant manner. So, instead of throwing away the coefficients like in Method-1, if one can *decrease the amplitude* of these coefficients which represent *mostly* noise (but not pure noise, they also have some musical content, throwing away these coefficients may cause loss of musically meaningful data) to an extent so that the noise will be *masked* by the musical content at that instant, then the music level would be significantly increased which will cause the signal to be more pleasant to the ear. The process uses an *overlapping windowed* version of the original signal. Mostly, a Hanning window of length 1024 with an overlap of 512 samples is used.

The method is described as follows:

1. Decompose the *windowed* signal into sub-signals using Wavelet Transform.

   This phase is the same as Step 1 in Method-1.

2. Select the sub-signals which have relatively high energy and reject the others.

   This phase is the same as Step 2 in Method-1.

3. Select the sub-signals whose energy is below a threshold value and shape their wavelet coefficients via an envelope function obtained from the sub-signal.

This process corresponds to the *Wavelet Coefficient Shaper* block in Figure 4.3 In this stage, the correlation between the wavelet coefficients and the samples of the sub-signal is used. For each wavelet coefficient, there corresponds a number of samples in the sub-signal. The number of coefficients depends on the resolution level of the sub-signal. For example, if the signal is x samples long and if the sub signal is at resolution level n, then $x/2^n$ samples correspond to each wavelet coefficient. Figure 4.2 shows an illustration of this partitioning. Then , the square root of the sum of the squares of the corresponding samples is calculated and the maximum of these numbers is normalised to 1. So, there corresponds a calculated number (can be called a normalised weight) for each wavelet coefficient. As a last step, the numbers are multiplied with the corresponding wavelet coefficients. So ,the coefficients which are already small (deemed noise) are shrinked (but not rejected). This helps the preservation of the auditory content when it coexists too much with noise.

4. Reconstruct the sub-signals from the shaped wavelet coefficients and add them with the sub-signals coming directly from Step II.

This process corresponds to the *Inverse Wavelet Transformer* and *Sub-Signal Adder* blocks in Figure 4.3. The signal obtained after the summation corresponds to the denoised signal.

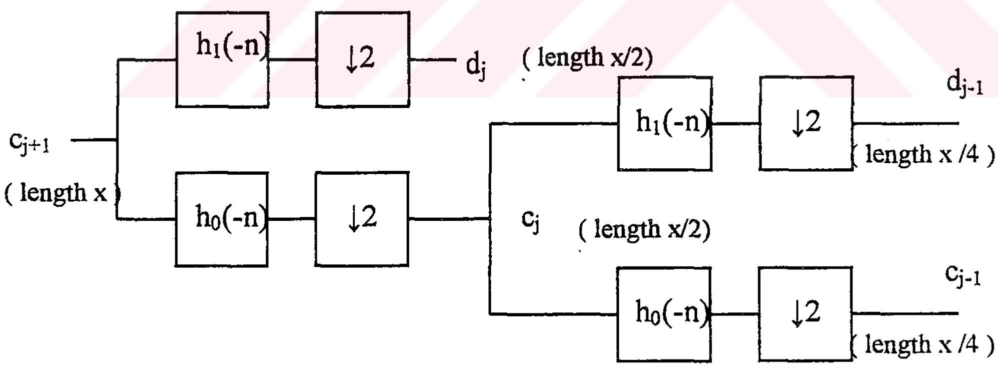


Figure 4. 2. Length of the coefficients at each resolution up to 2 decompositions

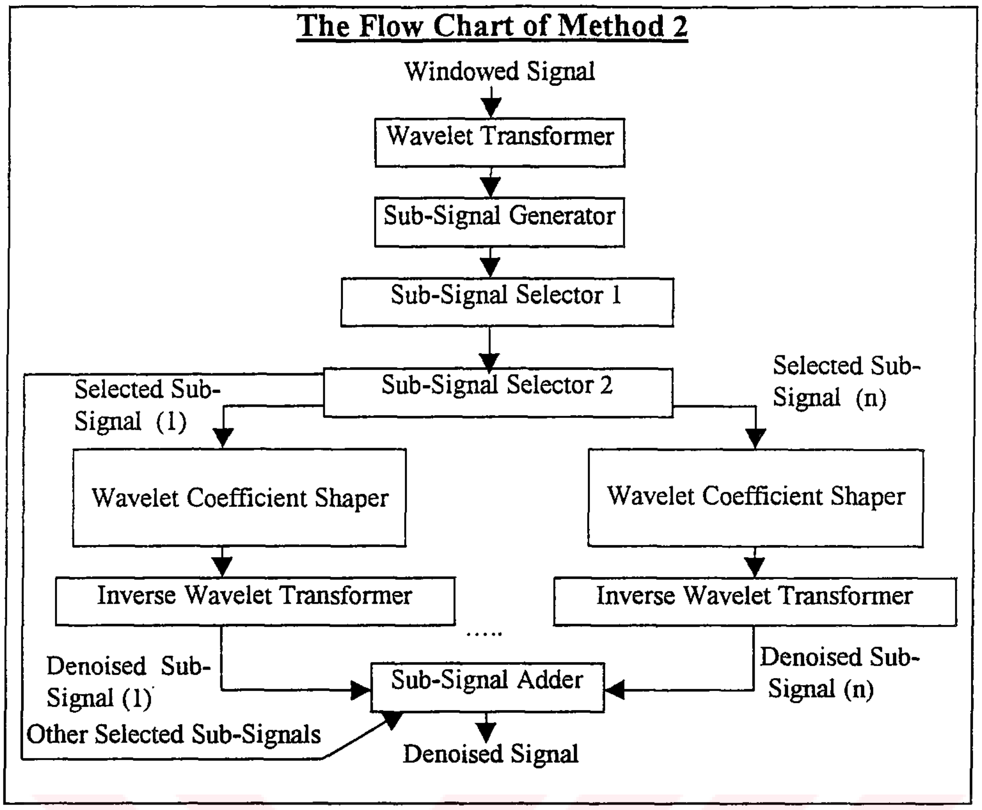


Figure 4. 3. The flow chart of the Method-2

4.4 Method-3

Method-3 is very similar to the entropy based denoising method introduced by Berger, Goldberg and Wickerhauser[9]. It is based on wavelet packet analysis and entropy based best basis selection, which was described in Chapter-3 in details. Again, the *windowed* versions of the signal is used and the steps that will be explained in the following pages assume operations being made on a *windowed* portion of the signal. The window length is mostly 1024 or 2048. The windows overlap by half the size to get rid of unwanted boundary effects.

The steps of method-3 are as follows:

1. Decompose the *windowed* signal using *wavelet packet decomposition*.

This process corresponds to the *Wavelet Packet Transformer* block in Figure 4.4. Wavelet packet decomposition has been explained briefly in Chapter-2. The resolution level is (*log2* of the maximum number of partitions in the signal space) is arbitrary and left to the implementor. In the experiments, it is mostly taken as 2,3 and 4, where 4 is very much enough for a signal of length 1024 or 2048. A resolution level of 4 means 16 partitions in the signal space. It must be noted that all the partitions have the same width in the frequency space, eg., the frequency space is split into 16 equal intervals for a resolution level of 4.

Mostly, *'Daubechies'* mother wavelet is used in the experiments. Also, other mother wavelets could be chosen. It is also left to the user.

2. Find the *best basis* which represents the windowed signal with the *least entropy concept* as the means of cost.

This process corresponds to the *Best Basis Selector* block in Figure 4.4. Note that for the selection of the best basis, the Normalized Shannon Entropy, which was explained in Chapter-3, is used as the cost function. The process of selecting the basis was also explained in Chapter-3.

3. Construct a wavelet packet structure which only contains the coefficients of the best basis selected in Step-2.

This process corresponds to the *Best Basis Packet Tree Constructor* block in Figure 4.4. This step is just zeroing the coefficients of the wavelet packets which are not kept as the best basis.

4. Find the *normalized node entropy* of each node in the best basis tree and select the node with the minimum (or maximum of the negated entropy) entropy.

This process corresponds to the *Normalized Node Entropy Calculator and Minimum Entropy Selector* block in Figure 4.4. It selects the best basis with the least entropy, which also means the basis with the highest information content.

5. Synthesise the windowed signal back from the basis selected in Step-4. This is the *windowed-synthesised signal.*

This process corresponds to the *Signal Synthesiser* block in Figure 4.4. The biggest difference between the method introduced by Berger, Wickerhauser and Goldberg [9], which was explained briefly in Chapter-3 and method-3 introduced here, is in this step. For reminding, it must be noted that the method in [9] has a normalised entropy threshold, which selects a *group* of basis among some selected *best basis.* In this method, only the basis with the best entropy is selected. So, only the basis which represents the *most coherent part* is taken into account.

6. Do steps 1-5 until all the input signal is windowed and processed and obtain a *synthesised signal.*
7. Subtract the synthesised signal from the input signal and consider the result of subtraction as the new input signal. Also, add the synthesised signal to the final denoised signal.

This step corresponds to the *Signal Subtractor* block in Figure 4.4.

8. Do steps 1-7 again until the total number of iterations is satisfied.

In this method, the total number of iterations totally depend on the choice of the user. In the experiments done, mostly 3 or 4 iterations was sufficient to obtain a denoised signal.

At each iteration, a signal which has a musical content is *peeled away* from the original signal. So, after some iterations, the remaining signal is mostly the noise with a very little musical content in it.

The biggest advantage of this method over the method introduced by Berger [9] is the lack of the *normalised entropy threshold*, which can be difficult to determine, or not good to select as a constant value. By taking the basis with the best entropy value, only the most coherent part of the signal (which is deemed as the musical content) is taken. Eventhough some musical content is left behind, it is not a problem. Because, it is then detected by the next iteration.

The selection of the number of iterations can be easily done by a listening test. This test can be done after each iteration. But it must be noted that, both the musical part and the remaining signal must be listened to, in order to make a decision that, the number of iterations taken up to that time is sufficient.

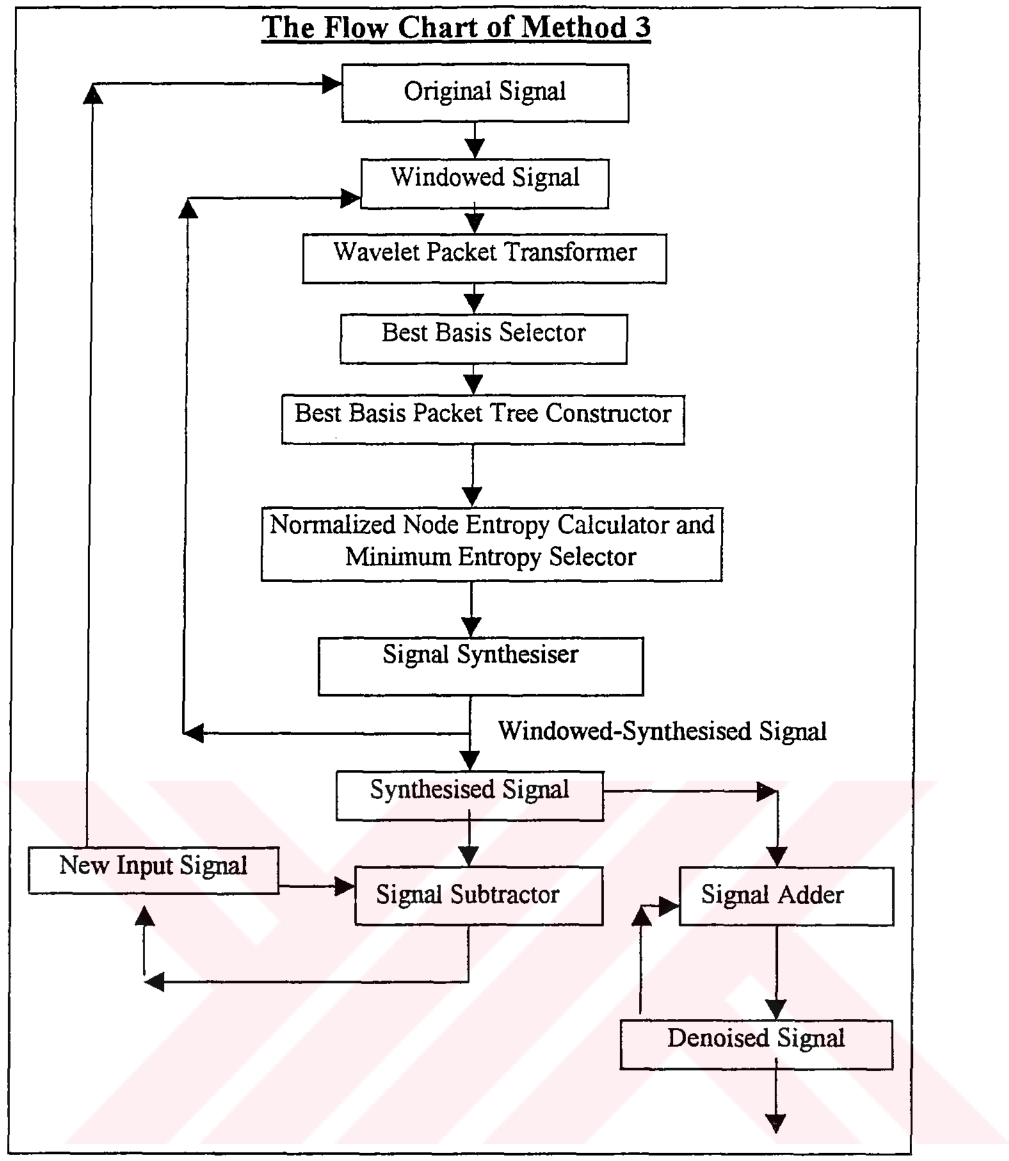


Figure 4. 4. The flow chart of the Method-3

4.5 The Experimental Results

In the experiments, an audio signal which was sampled from an old recording of famous Turkish art music singer Zeki Müren is used. The recording is sampled at 22050 Hz and its duration is 2.30 minutes, which is quite enough to understand the performances of the methods introduced.

Of course, just by looking at the graphics which will be shown below, it is not quite easy to decide which method works well. In order to have a better idea, listening tests have to be made.

The main wavelets used for the tests are Daubechies, Coiflet and Symmlet. For each method, the mother wavelet with the best performance is found by trial and error.

4.5.1 The Experimental Results of Method-1

In the results which will be given below, Daubechies mother wavelet is used. The thresholds for the *Sub-Signal Selector-1* , *Sub-Signal Selector-2* and *Audio Content Bearing Packets Selector* blocks are 0.05, 0.1 and 0.3 respectively. The window length is 1024 samples. Matlab source code which generates the following outputs is given in Appendix B with the name *method1.m*. As it was mentioned before, the success of the methods can be best understood via the listening tests. So, the figures below can only give a rough idea about the results.

The input signal is an old recording of a song of Zeki Müren. It is 2:30 minutes long and it will be used as input for the output representation of all the methods.

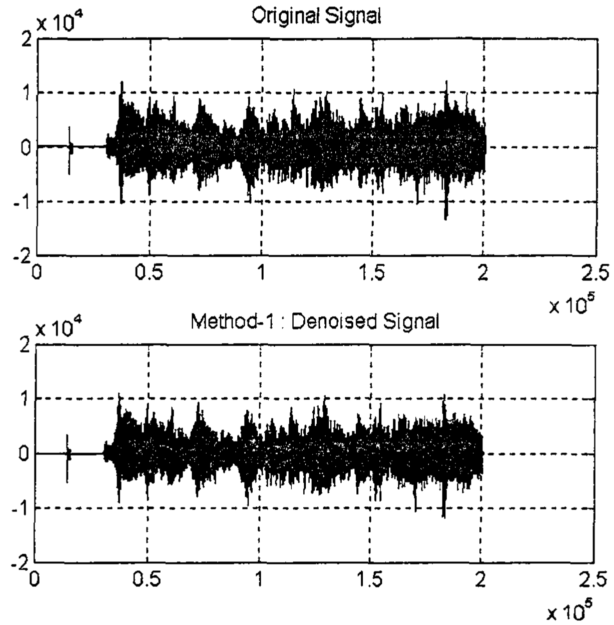


Figure 4. 5. The original signal and the denoised signal outputs of Method-1

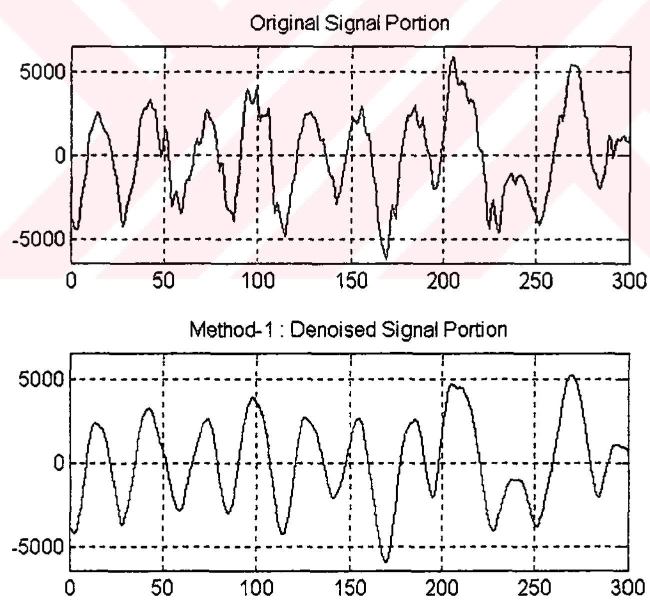


Figure 4. 6. A Portion of the original and the denoised signal of Method-1

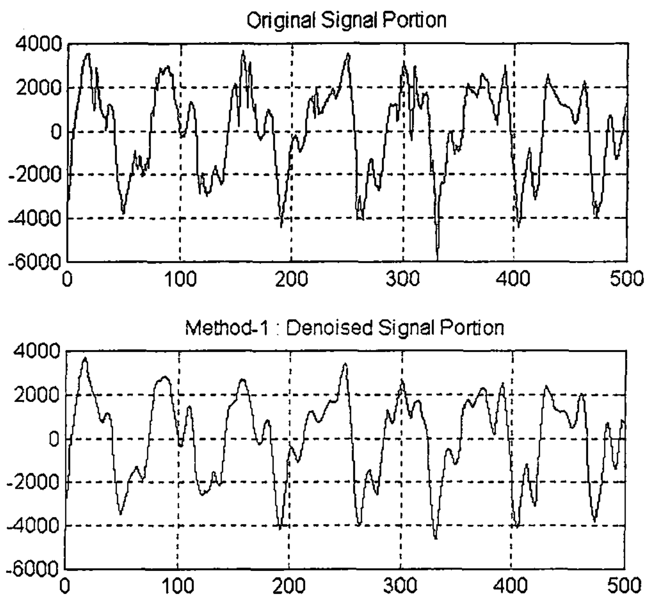


Figure 4. 7. Another portion of the original and the denoised signal of Method-1

4.5.2 The Experimental Results of Method-2

In the results which will be given below, Symmlet mother wavelet is used. The thresholds for the *Sub-Signal Selector-1* and *Sub-Signal Selector-2* are 0.007 and 0.17 respectively. The window length is 2048 samples. Matlab source code which generates the following outputs is given in Appendix B with the name *method2.m*.

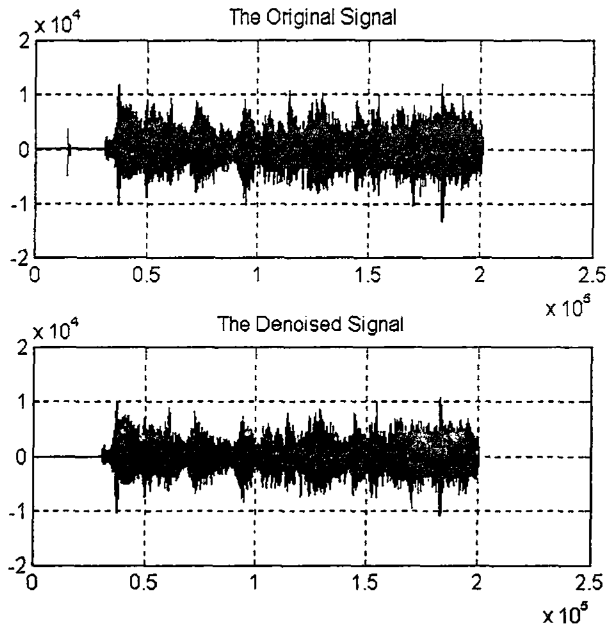


Figure 4. 8. The original and the denoised signal outputs of Method-2

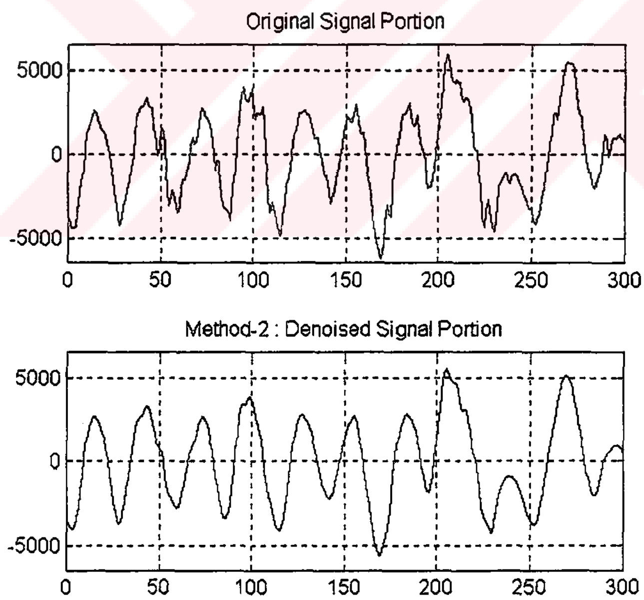


Figure 4. 9. A portion of the original and the denoised signal of Method-2

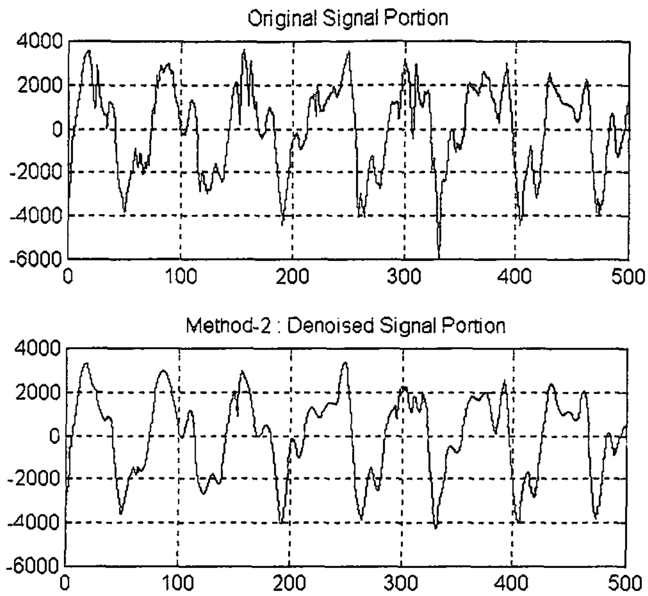


Figure 4. 10. Another portion of the original and the denoised signal of Method-2

4.5.3 The Experimental Results of Method-3

In the results which will be given below, Daubechies mother wavelet is used. The maximum resolution level for the *Best Basis Selector* block is 2. The algorithm is iterated twice. The window length is 1024 samples. Matlab source code which generates the following outputs is given in Appendix B with the name *method3.m*.

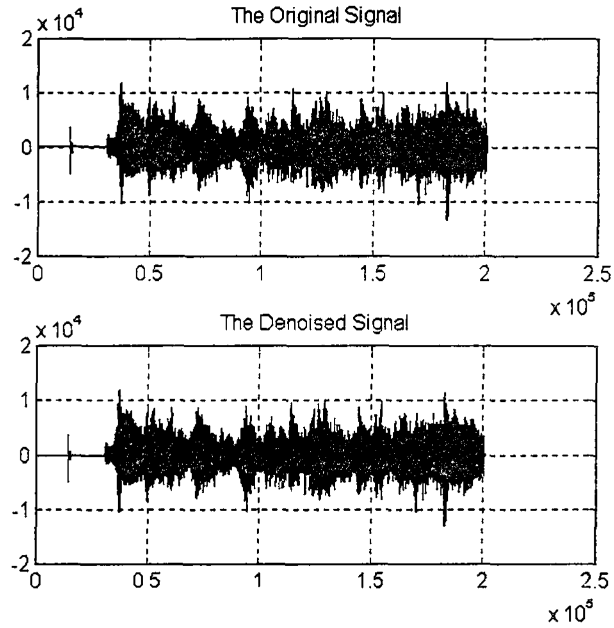


Figure 4. 11. The original and the denoised signal outputs of Method-3

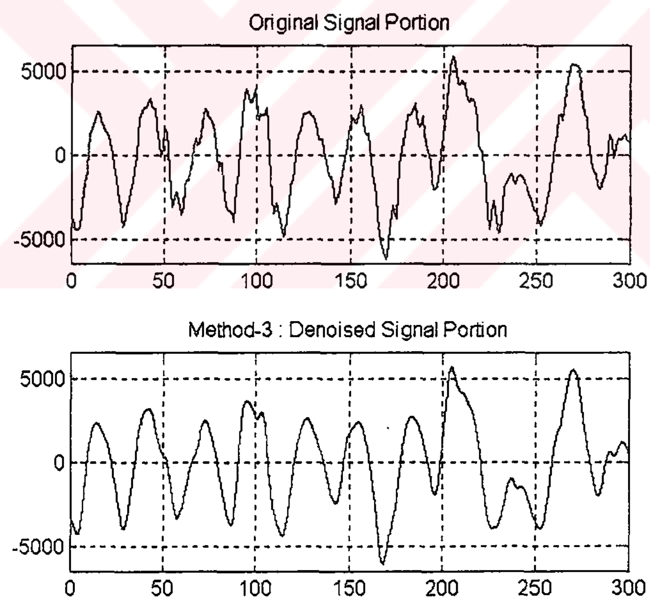


Figure 4. 12. A portion of the original and the denoised signal of Method-3

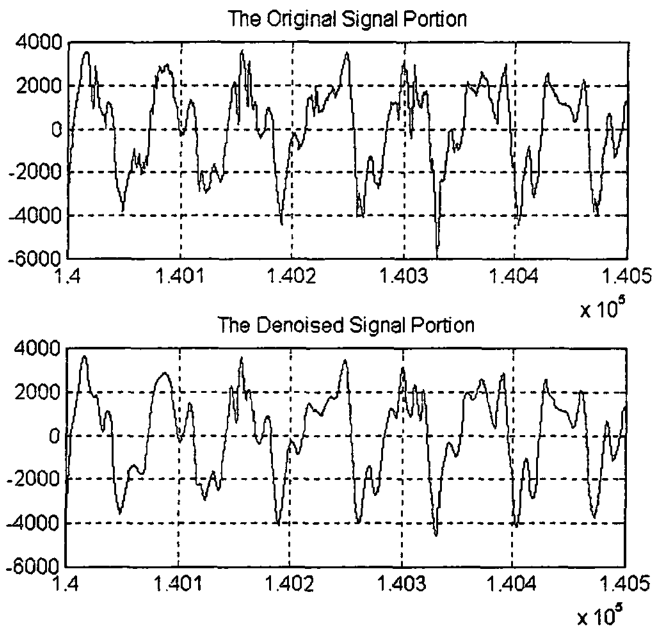


Figure 4. 13. Another portion of the original and the denoised signal of Method-3

It can be seen from the above figures that, when the noise in the signal is low (like in Figure 4.6 , Figure 4.9 and Figure 4.12) , the outputs of the methods are very much alike. They all have some sort of low-pass filtering characteristics. But when the noise in the signal is high, or the distortion is more (like in Figure 4.7, Figure 4.10 and Figure 4.13), then each method generate different waveforms.

According to the listening test, the performance of the methods can be ordered from the worst to the best as Method-1, Method-2 and Method-3.

CHAPTER 5

CONCLUSION

In this work, denoising of degraded audio signals using wavelet analysis is presented. Three methods have been proposed for this purpose. All of the methods are implemented on the *overlapped window* versions of the original signal with window lengths of 1024, 2048 and 4096 samples.

In Method-1, both the thresholding and the wavelet packet analysis tools are used [10]. In addition to these, some decisions are made according to the relative energy contents of the sub-signals obtained during the process. A deeper analysis on the sub-signals is done via the *wavelet packet* analysis, which provided a better resolution in both time and frequency. Some wavelet packets are then selected via a thresholding process for the synthesis of the denoised sub-signal. Then, the synthesised sub-signals and the ones selected before by another sub-signal selector process are added up to construct the denoised signal.

The most important characteristic of this method is the use of the wavelet packets for the analysis. The main problem is the selection of the audio bearing packets, which can be worked on in more detail as a future work.

Method-2 is based on the shaping of the wavelet coefficients by the help of the energies of the sub-signals[10]. The idea behind this process is to have some kind of a *non-linear filtering* approach , which is based on the energy content of the signal. No coefficients are thrown out like in thresholding scheme, but *decreased in amplitude* which means their contribution to the overall system is decreased. If one can shrink the coefficients which are mostly related to noise, then the output signal has a better performance in means of the suppression of noise.

The main disadvantage of the method is that, if the energy of the musical content in the sub-signal is low, then noise's energy is mostly dominant in the shaping process of the coefficients. How to get rid of this problem can be examined as a future work.

Method-3 is very similar to the entropy based method introduced by Berger, Wickerhauser and Goldberg [9]. It is based on the wavelet packet analysis and entropy based best basis selection process. This process can be taken into account as a method for the selection of the audio bearing packets which was introduced in Method-1. The algorithm works in a *controllable* iteration-based manner. The output is fed back to the input for a further denoising process if necessary.

The number of iterations must be carefully selected. If one selects a high number of iterations, then the algorithm starts selecting noise as the coherent part (musical content) of the input signal. The reason is that, at each iteration, the coherent part of the signal is *peeled off* from the original signal and this is done by an entropy based algorithm. After some iterations, the noise will be very much dominant in the input signal, which can cause the algorithm to choose the noise as the coherent part. So, one must be very careful in selecting the number of iterations. In this work, the number of iterations is decided by trial and error. An *adaptive* method for the control of the number of iterations can be worked on as a future work.

It must also be noted that, in denoising type algorithms, the results of the experiments can be best understood by the listening tests, because the plots of the outputs can be misleading. The listening tests show that, method-3 has the best performance floowed by method-2 and method-1.

As a future work, mother wavelet functions which represent musical instruments can be formed. By this way, the signal can be divided into *instrument based* bands, and each instrument can be analysed separately by their own wavelet functions. So, while denoising, the problem of throwing away extra musical information can be overcome.

REFERENCES

[1] Burrus C. S., Gopinath R. A., Guo H. , " Introduction to Wavelets and Wavelet Transforms", *Prentice Hall, Upper Saddle River, NJ,* 1998.

[2] Daubechies I., " Ten Lectures on Wavelets" , *SIAM, Philadelphia* , PA, 1992.

[3] Sweldens W. , " Wavelets: What next? ", *Proceedings of the IEEE,* 84(4):680-685, April 1996.

[4] Mallat S.G. , " A Theory of Multiresolutional Signal Decomposition: The Wavelet Representation " , *IEEE Transactions on Pattern Recognition and Machine Intelligence,* 11(7): 674-693, July 1989.

[5] Vaidyanathan P.P., " Multirate Systems and Filter Banks ", *Prentice Hall, Englewood Cliffs, NJ,* 1992.

[6] Burrus C.S., Gopinath R.A., " Introduction to Wavelets and Wavelet Transforms: A Tutorial ", IEEE *ICASSP 93, Minnesota,* April 1993

[7] Donoho D.L., " Denoising by Soft Thresholding ", *IEEE Transactions on Information theory,* 41(3):613-627, May 1995.

[8] Donoho D.L., Johnstone I.M., " Ideal Spatial Adaptation via Wavelet Shrinkage ", *Biometrica,* 81:425-455, 1994.

[9] Berger J., Coifman R.R., Goldberg M.J., " Removing Noise From Music Using Local Trigonometric Bases and Wavelet Packets ", *Journal of Audio Engineering Society,* Vol.42, No.10:808-818, October 1994.

[10] Aksu E.B., Çiloğlu T., Demirekler M., "Dalgacık İncelemesi İle Yıpranmış Müzik İşaretlerinin Onarılması", *SIU98 Bildirimler Kitabı,* Cilt II , 480-486, Mayıs 1998.

[11] Vaidyanathan P.P., Djokovic I., " Wavelet Transforms ", *The Circuits and Systems Handbook*, Chapter 6, 134-219, CRC Press and IEEE Press, Roca Raton, 1995.

[12] Smith M.J., Barnwell T.P., " Exact Reconstruction Techniques for Tree-Structured Subband Coders ", *IEEE Transactions on Acoustics, Speech ad Signal Processing,* 34: 434-441, June 1986.

[13] Coifman R.R., " Wavelet Analysis and Signal Processing ", *Signal Processing Theory,* 59-68, Springer-Verlag, NY, 1990

[14] Donoho D.L., " Nonlinear Wavelet Methods for Recovery of Signals, Densities, and Spectra from Indirect and Noisy Data ", *Different Perspectives on Wavelets* , 173-205, American Mathematical Society, Providence, 1993.

[15] Donoho D.L., Johnstone I.M., Kerkyacharian G., Picard D., " Wavelet Shrinkage: asymptopia? ", *Journal Royal Statistics Society Ser B.*, 57(2):301-337, 1995.

[16] Donoho D.L., Johnstone I.M., " Ideal Denoising in an Orthonormal Basis Chosen From a Library of Bases ", *Stanford Statistics Dept.*, Report 461, September 1994.

[17] Johnstone I.M., SilvermanB.W., " Wavelet Threshold Estimators for Data with Correlated Noise ", *Technical Report, Statistics Dept. , University of Bristol*, September 1994.

[18] Coifman R.R., Wickerhauser M.V., " Wavelets and Adapted Waveform Analysis. A Toolkit for Signal Processing and Numerical Analysis", *Proc. Symp. In Applied Mathematics*, vol. 47, 119-153, 1993.

[19] Vaseghi S.V., "Algorithms for Restoration of Archived Recordings", *PHD thesis*, University of Cambridge, 1988

[20] Godsill S.J., "Algorithms for the Reconstruction of Degraded Audio", *PHD thesis*, University of Cambridge,1993

[21] Godsill S.J., Rayner J.W., "A Bayesian Approach to the Restoration of Degraded Audio Signals," *IEEE Transactions on Speech and Audio Processing*, Vol.3,No. 4, July 1995.

[22] Godsill S.J., "Bayesian Enhancement of Speech and audio Signals In the Presence of Both Impulsive and Background Noise", 1997.

[23] Hicks C.H., Godsill S.J., " A Two Channel Approach to the Removal Of Impulsive Noise from Archived Recordings," *IEEE ICASSP Proceedings*, 1994

[24] Godsill S.J., " Recursive Restoration of Pitch Variation Defects in Musical Recordings", IEEE *ICASSP Proceedings*, 1994

[25] Cappe O., "Enhancement of Musical Signals Degraded by Background Noise Using Long-Term Behaviour of the Short -Term Spectral Components", *IEEE ICASSP Proceedings*, 1993.

[26] Berger J., Coifman R., Goldberg M., "A two Stage Adaptive Process to Remove Noise From Music In a Recorded Audio Signal", *Yale Center For Studies in Music Technology*, 1995.

[27] Berger J., Coifman R., Goldberg M., "A Method of Removing Noise from Old Recordings". *Proceedings of the 1994 International Computer Music Conference*, 1994

APPENDIX A

NECESSARY CONDITIONS AND PROPERTIES OF SCALING FUNCTIONS AND WAVELETS

A.1 Introduction

In Appendix A, the necessary conditions and the properties of the scaling function and the wavelet are given [1]. The properties of $\varphi(t)$ will be examined by considering the equation of which it is a solution. The basic recursion equation that comes from the multiresolutional formulation is

$$\varphi(t) = \sum_n h(n)\sqrt{2}\varphi(2t-n) \tag{A.1}$$

with *h(n)* being the scaling coefficients and $\varphi(t)$ being the scaling function which satisfies equation (A.1). Now, some necessary conditions for the equation to possess will be considered. The brief proofs of the theorems and conditions which will be stated here can be found in [1].

A.2 Necessary Conditions

Theorem 1: If $\varphi(t) \in L^1$ is a solution of the basic recursion equation (A.1) and if $\int \varphi(t)dt \neq 0$, then

$$\sum_n h(n) = 2 \tag{A.2}$$

The proof of this theorem requires only an interchange in the order of a summation and integration (which is allowed in L^1) but no assumption on the orthogonality of the basis functions or any other properties of $\varphi(t)$ other than the nonzero integral.

This theorem shows that, unlike linear constant coefficient differential equations, not just any set of coefficients will support a solution. The coefficients must satisfy the linear equation (A.2). This is the weakest condition on *h(n)*.

Theorem 2: If $\varphi(t)$ is an L^1 solution to the basic recursion equation (A.1) with $\int \varphi(t)dt = 1$, and

$$\sum_l \varphi(t-l) = \sum_l \varphi(l) = 1 \qquad \text{(A.3)}$$

with the frequency domain representation of $\varphi(t)$, $\Phi(\pi+2\pi k) \neq 0$ for some k, then

$$\sum_n h(2n) = \sum_n h(2n+1) \qquad \text{(A.4)}$$

where equation (A.3) may have to be a distributional sum. Conversely, if (A.4) is satisfied, than (A.3) is true.

Equation (A.4) is called the *fundamental condition*, and it is weaker than requiring orthogonality but stronger than (A.2). It is simply a result of requiring the equations resulting from evaluating (A.1) on the integers to be consistent. Equation (A.3) is also called the *partitioning of unity*.

Theorem 3: If $\varphi(t)$ is an $L^2 \cap L^1$ solution to (A.1) and if integer translates of $\varphi(t)$ are orthogonal as defined by

$$\int \varphi(t)\varphi(t-k)dt = E\delta(k) = \begin{cases} E & if \quad k=0 \\ 0 & otherwise \end{cases} \tag{A.5}$$

then

$$\sum_n h(n)h(n-2k) = \delta(k) = \begin{cases} 1 & if \quad k=0 \\ 0 & otherwise \end{cases} \tag{A.6}$$

Notice that this does not depend on the normalisation of $\varphi(t)$. If $\varphi(t)$ is normalised by the square root of its energy $\sqrt{E}$, then integer translates of $\varphi(t)$ are orthonormal defined by

$$\int \varphi(t)\varphi(t-k)dt = \delta(k) = \begin{cases} 1 & if \quad k=0 \\ 0 & otherwise \end{cases} \tag{A.7}$$

This theorem shows that in order for the solutions of (A.1) to be orthogonal under integer translation, it is necessary that the coefficients of the recursive equation be orthogonal themselves after decimating or downsampling by two. If $\varphi(t)$ and/or *h(n)* are complex functions, complex conjugation must be used in (A.5), (A.6) and (A.7).

Coefficients *h(n)* that satisfy (A.6) are called *quadrature mirror filter* (QMF) or *conjugate mirror filter* (CMF), and the condition (A.6) is called the *quadratic condition.*

Corollary 1: Under the assumptions of Theorem 3, the *norm* of *h(n)* is unity.

$$\sum_n |h(n)|^2 = 1 \tag{A.8}$$

Not only must the sum of *h(n)* equal $\sqrt{2}$, but for orthogonality of the solution, the sum of the squares of *h(n)* must be one, both independent of any normalisation of $\varphi(t)$. This unity normalisation of *h(n)* is the result of the $\sqrt{2}$ term in (A.1).

Corollary 2: Under the assumptions of Theorem 3,

$$\sum_n h(2n) = \sum_n h(2n+1) = \frac{1}{\sqrt{2}} \qquad \text{(A.9)}$$

The meaning of this equation is that, the individual sums of the even and odd terms in *h(n)* must be $1/\sqrt{2}$, independent of any normalisation of $\varphi(t)$. Although stated here as necessary orthogonality, the results hold under weaker non-orthogonal conditions as is stated in Theorem 2.

Theorem 4: If $\varphi(t)$ has *compact support* on $0 \le t \le N-1$ and if $\varphi(t-k)$ are *linearly independent*, then *h(n)* also has compact support over $0 \le n \le N-1$, i.e,

$$h(n) = 0 \quad for \quad n < 0 \quad and \quad n > N-1 \qquad \text{(A.10)}$$

Thus, N is the length of the *h(n)* sequence. If the translates are not independent, one can have *h(n)* with infinite support while $\varphi(t)$ has finite support.

These theorems state that, if $\varphi(t)$ has compact support and is orthogonal over integer translates, $N/2$ bilinear or quadratic equations (A.6) must be satisfied in addition to the one linear equation (A.2). The support or length of *h(n)* is *N*, which must be an *even* number. The number of degrees of freedom in choosing these *N* coefficients is then $N/2-1$.

A.3 Frequency Domain Necessary Conditions

Now, the frequency domain versions of the necessary conditions for the existence of $\varphi(t)$ will be introduced. For more details on this section, one can look at [11].

The Fourier transform of $\varphi(t)$ is defined as

$$\Phi(\omega) = \int_{-\infty}^{\infty} \varphi(t) e^{-j\omega t} dt \tag{A.11}$$

and the discrete time Fourier transform of *h(n)* is defined as

$$H(\omega) = \sum_{n=-\infty}^{\infty} h(n) e^{-j\omega n} \tag{A.12}$$

by the help of (A.1) ,Φ(ω) can also be defined as

$$\Phi(\omega) = \frac{1}{\sqrt{2}} H(\omega/2) \Phi(\omega/2) \tag{A.13}$$

which after iterations becomes

$$\Phi(\omega) = \prod_{k=1}^{\infty} \left\{ \frac{1}{2} H\left(\frac{\omega}{2^k} \right) \right\} \Phi(0) \tag{A.14}$$

Note that the above two equations hold if $\sum_n h(n) = \sqrt{2}$ and $\Phi(0)$ is well defined.

Theorem 5: If $\varphi(t)$ is a L^1 solution of the basic recursion equation (A.1), then the following equivalent conditions must be true:

$$\sum_n h(n) = H(0) = \sqrt{2} \tag{A.15}$$

This states that the basic existence requirement (A.2) is equivalent to requiring that the FIR filter's frequency response at DC (*w*=0) be $\sqrt{2}$.

Theorem 6: For $h(n) \in l^1$, then

$$\sum_n h(2n) = \sum_n h(2n+1) \qquad \text{if and only if } H(\pi) = 0 \qquad \text{(A.16)}$$

which says that the frequency response of the FIR filter with the impulse response *h(n)* is zero at the *Nyquist Frequency* (w= π).

Theorem 7: If $\varphi(t)$ is a solution to (A.1) in $L^2 \cap L^1$ and Φ(w) is a solution of (A.13), such that Φ(0)≠0, then

$$\int \varphi(t)\varphi(t-k)dt = \delta(k) \quad \text{if and only if} \quad \sum_l \left|\Phi(\omega + 2\pi l)\right|^2 = 1 \qquad \text{(A.17)}$$

This is a frequency domain equivalent to the time domain definition of orthogonality of the scaling function. It allows applying the orthonormal conditions to frequency domain arguments. It also gives an insight into just what time domain orthogonality requires in the frequency domain.

Theorem 8: For any $h(n) \in l^1$,

$$\sum_n h(n)h(n-2k) = \delta(k) \text{ if and only if } \left|H(\omega)\right|^2 + \left|H(\omega+\pi)\right|^2 = 2 \qquad \text{(A.18)}$$

This theorem gives the equivalent time and frequency domain conditions on the scaling coefficients and states that the orthogonality requirement (A.6) is equivalent to the FIR filter with *h(n)* as coefficients being what is called a Quadrature Mirror Filter (QMF) [12]. Note that (A.15), (A.16) and (A.18) require $\left|H(\pi/2)\right| = 1$ and that the filter is a *half band* filter.

A.4 The Wavelet

The wavelet is defined as follows:

$$\psi(t) = \sum_n h_1(n)\sqrt{2}\varphi(2t-n) \qquad n \in Z \qquad \text{(A.19)}$$

where $h_1(n)$ is

$$h_1(n) = (-1)^n h(N-1-n) \qquad \text{(A.20)}$$

Theorem 9: If the scaling coefficients *h(n)* satisfy the conditions for existence and orthogonality of the scaling function and the wavelet is defined by (A.19), then the integer translates of this wavelet span W_0, the orthogonal component of V_0, both being in V_1 (see Figure 2.5), i.e., the wavelet is orthogonal to the scaling function at the same scale,

$$\int \varphi(t-n)\psi(t-m)dt = 0 \qquad \text{(A.21)}$$

if and only if the coefficients $h_1(n)$ are given by (A.20).

Theorem 10: If Theorem 9 is true, then

$$\sum_n h(n)h_1(n-2k) = 0 \qquad \text{(A.22)}$$

The translation orthogonality and the scaling function-wavelet orthogonality conditions in (A.6) and (A.22) can be combined to give

$$\sum_n h_l(n)h_m(n-2k) = \delta(k)\delta(l-m) \qquad \text{(A.23)}$$

if $h_0(n)$ is defined as *h(n)*.

Theorem 11: If *h(n)* satisfies the linear and quadratic admissibility conditions of (A.2) and (A.6), then

$$\sum_n h_1(n) = H_1(0) = 0 \qquad \text{(A.24)}$$

$$|H_1(\omega)| = |H(\omega+\pi)| \tag{A.25}$$

$$|H(\omega)|^2 + |H_1(\omega)|^2 = 2 \tag{A.26}$$

and

$$\int \psi(t)dt = 0 \tag{A.27}$$

The wavelet is usually scaled so that its norm is unity.

The results in this section have not included the effects of *integer shifts* of the scaling function or the wavelet coefficients $h(n)$ or $h_1(n)$. In a particular situation, these sequences may be shifted to make the corresponding FIR filter causal.

A.5 Further Properties of the Scaling function and Wavelet

The scaling function and the wavelet have some remarkable properties that should be examined in order to understand wavelet analysis and to gain some intuition for these systems. Likewise, the scaling and wavelet coefficients have important properties that should be considered.

The basic recursive equation for the scaling function, defined in (A.1) is homogenous, so its solution is unique only within a normalisation factor. In most cases, both the scaling function and the wavelet are normalised to unit energy or unit norm. In the properties discussed here, the energy is defined as $E = \int |\psi(t)|^2 dt = 1$. Other normalisations can easily be used if desired.

A.5.1 General Properties not Requiring Orthogonality

There are several properties that are simply a result of the multiresolutional equation (A.1) and, therefore, hold for orthogonal systems.

Property 1: The normalisation of $\varphi(t)$ is arbitrary and is given as E. e is usually set to 1 so that the basis functions are orthonormal and coefficients can easily be calculated with inner products.

Property 2: Not only can the scaling function be written as a weighted sum of functions in the next higher scale space as stated in the basic recursion equation (A.1), but it can also be expressed in higher resolution spaces:

$$\varphi(t)=\sum_{n} h^{(j)}(n)2^{j/2}\varphi(2^{j}t-n) \tag{A.28}$$

where $h^{(1)}(n)=h(n)$ and for $j \geq 1$

$$h^{(j+1)}(n)=\sum_{k} h^{(j)}(k)h^{(j)}(n-2k) \tag{A.29}$$

Property 3: A formula for the dyadic sum of the samples of $\varphi(t)$ is

$$\sum_{k}\varphi\left(\frac{k}{2^{j}}\right)=2^{j} \tag{A.30}$$

Property 4: A *partition of unity* follows from (A.30) for j=0 is

$$\sum_{m}\varphi(m)=1 \tag{A.31}$$

Property 5: A generalised partition of unity exists if $\varphi(t)$ is continuous

$$\sum_{m}\varphi(t-m)=1 \tag{A.32}$$

Property 6: A frequency domain statement of the basic recursion equation (A.1) is

$$\Phi(\omega)=\frac{1}{\sqrt{2}}H(\omega/2)\Phi(\omega/2) \tag{A.33}$$

Property 7: Successive approximations in the frequency domain is often easier to analyse than the time domain version in (A.1). The convergence properties of this infinite product are very important.

$$\Phi(\omega)=\prod_{k=1}^{\infty}\left\{\frac{1}{2}H\left(\frac{\omega}{2^k}\right)\right\}\Phi(0) \qquad \text{(A.34)}$$

A.5.2 Properties that Depend on Orthogonality

The following properties depend on the orthogonality of the scaling and wavelet functions.

Property 8: The square of the integral of $\varphi(t)$ is equal to the integral of the square of $\varphi(t)$.

$$\left[\int\varphi(t)dt\right]^2=\int\varphi(t)^2dt \qquad \text{(A.35)}$$

Property 9: The integral of the wavelet is necessarily zero.

$$\int\psi(t)dt=0 \qquad \text{(A.36)}$$

Property 10: not only are integer translates of the wavelet orthogonal, different scales are also orthogonal.

$$\int 2^{j/2}\psi\left(2^jt-k\right)2^{i/2}\psi\left(2^it-l\right)dt=\delta(k-l)\delta(k-i) \qquad \text{(A.37)}$$

where the norm of $\psi(t)$ is one.

Property 11: The scaling function and the wavelet are orthogonal over both scale and translation.

$$\int 2^{j/2}\psi\left(2^jt-k\right)2^{i/2}\varphi\left(2^it-l\right)dt=0 \qquad \text{(A.38)}$$

for all integer i,j,k,l where j ≤ i.

Property 12: a frequency domain statement of the orthogonality requirements is in (A.5). It also is a statement of equivalent energy measures in the time and frequency domains as in Parseval's theorem, which is true with an orthogonal basis set.

$$\sum_k |\Phi(\omega + 2\pi k)|^2 = \int |\Phi(\omega)|^2 d\omega = \int |\varphi(t)|^2 dt = 1 \qquad \text{(A.39)}$$

Property 13: The scaling coefficients can be calculated from the orthogonal or tight frame scaling functions by

$$h(n) = \sqrt{2} \int \varphi(t)\varphi(2t - n)dt \qquad \text{(A.40)}$$

Property 14: the wavelet coefficients can be calculated from the orthogonal or tight frame scaling functions by

$$h_1(n) = \sqrt{2} \int \psi(t)\varphi(2t - n)dt \qquad \text{(A.41)}$$

APPENDIX B

MATLAB SOURCE CODES

B.1 Method1.m

```
%*********************************************************************
%                          MS Thesis Source Code -1
% Name : method1.m
% By: Emre Baris Aksu
% Creation Date : 7/3/1998
% Last Update   : 18/11/1998
%*********************************************************************
%
% Descripton:
%         This code is the implementation of Method-1. It uses Wavelab v7.00 library.


clear;
% Read the input signal
fid = fopen('zmg.raw' , 'r');
indata = fread(fid , 'int16');
fclose(fid);
fs = 22050;

length_indata = length(indata);
window_length = 1024;
step_length = window_length / 2;
maxlevel = log2(window_length) - 1;

energy_threshold_2 = 0.05;
energy_threshold_3 = 0.1;
energy_threshold_4 = 0.001;

%*** The Hanning window of length window_length
han_window = hanning(window_length);
L = 1; %*** The coarsest level of decomposition...
D = 4; %*** The maximum depth in packet analysis...
%*** number of packet coefficients at the maximum depth.
packet_length = length(packet(D , 0 , window_length));

%*** The Quadrature Mirror Filter coefficients ...
qmf = makeonfilter('Daubechies' , 12);
btree = zeros(1,2^(D+1) - 1);
%*** Selecting all possible nodes at the maximum depth of packet analysis...
btree(1:2^D - 2) = ones(1,2^D - 2);

denoised_data = zeros(1, length_indata);
```

```
%*** MAIN LOOP BEGINS...

for wcounter = 0 : floor(length_indata / step_length) - 2
%*** Display the counter
  wcounter

%*** Multiply the input data with the Hanning Window...
  wdata = indata(wcounter * step_length + 1  : wcounter * step_length + window_length )
.* han_window;

%*** START OF STEP 1
%*** Find the wavelet transform of the windowed signal...
  wc = fwt_po(wdata,L,qmf);

%*** Decompose the signal into sub-signals...
  for j = 0 : log2(window_length) - 1,
    eval(['di' int2str(j) ' = wc(dyad(' int2str(j) '));']);
    eval(['wc' int2str(j) ' = zeros(1,window_length);']);
          eval(['wc' int2str(j) ' (dyad(' int2str(j) ')) = di' int2str(j) ';']);
    eval(['y' int2str(j) ' = iwt_po(wc' int2str(j) ', L , qmf);']);
  end;
%*** END OF STEP 1...

%*** START OF STEP 2...

  signal_energy = sum(wdata .* wdata);
  for j = 1 : log2(window_length) - 1,
    eval(['sub_signal_energy(' int2str(j) ') = sum(wc' int2str(j) '.* wc' int2str(j) ');']);
          sub_signal_energy(j) = sub_signal_energy(j) / signal_energy;
          if sub_signal_energy(j) < energy_threshold_2
            eval(['wc' int2str(j) '(dyad(' int2str(j) ')) = zeros(1 , length(dyad(' int2str(j) ')));']);
      eval(['y' int2str(j) ' = iwt_po(wc' int2str(j) ', L , qmf);']);
          end;
  end;
%*** END OF STEP 2...

%*** START OF STEP 3 and 4...
  for j = 1 : log2(window_length) - 1,
          if sub_signal_energy(j) < energy_threshold_3
            if sub_signal_energy(j) > energy_threshold_2
              eval(['wp' int2str(j) ' = wpanalysis(y' int2str(j) ' , D , qmf);']);

                    for k = 1 : D,
                      eval(['wp' int2str(j) '(: , k) = zeros(window_length , 1);']);
                    end;

                    for k = 0 : 2^D - 1,
                      eval(['packet_energies(k+1) = sum(wp' int2str(j) '(packet(D , k ,
                      window_length) , D + 1) .* wp' int2str(j) '(packet(D , k ,
                      window_length), D + 1));']);
                    end;

                    packet_energies = packet_energies / sum(packet_energies);

                    for k = 0 : 2^D - 1,
                      if packet_energies(k+1) < energy_threshold_4
```

```
                    eval(['wp' int2str(j) '(packet(D , k , window_length) , D + 1) =
                    zeros(packet_length , 1);']);
                  end;
                end;
          eval(['y' int2str(j) ' = wpsynthesis(btree , wp' int2str(j) ', qmf);']);
                eval(['wc' int2str(j) ' = fwt_po(y' int2str(j) ', L , qmf);']);
   end;
  end;
 end;
%*** END OF STEP 3 and 4...

%*** START OF STEP 5...
 wc_denoised = zeros(1, window_length);
 for j = 1 : log2(window_length) - 1
  eval(['wc_denoised = wc_denoised + wc' int2str(j) ';']);
 end;
 denoised_wdata = iwt_po(wc_denoised , L , qmf);

 if wcounter ==0
  denoised_data( 1 : window_length) = denoised_wdata;
 else
  denoised_data(wcounter * step_length + 1 : (wcounter + 1) * step_length) =
denoised_data(wcounter * step_length + 1 : (wcounter +1) * step_length) +
denoised_wdata(1 : step_length);
  denoised_data((wcounter + 1) * step_length + 1 : (wcounter + 2) * step_length) =
denoised_wdata(step_length + 1 : 2 * step_length);
 end;
%*** END OF STEP 5...
% plot(wdata);hold on;plot(denoised_wdata,'r');hold off;pause;zoom on;
%*** END Of MAIN LOOP...
end;

fid = fopen('d:\matlab42\dist\zmgmet1.raw', 'w');
fwrite(fid,denoised_data,'int16');
fclose(fid);
```

B.2 Method2.m

```
%**********************************************************************
%                          MS Thesis Source Code -2
% Name : method2.m
% By: Emre Baris Aksu
% Creation Date : 11/3/1998
% Last Update   : 18/11/1998
%**********************************************************************
%
% Descripton:
%         This code is the implementation of Method-2. It uses Wavelab v7.00 library.
%         and also calls another function called wc_shape().


clear;
% Read the input signal
fid = fopen('zmg.raw' , 'r');
indata = fread(fid , 'int16');
fclose(fid);

fs = 22050;
length_indata = length(indata);
L = 1;
window_length = 4096;
step_length = window_length / 2;
maxlevel = log2(window_length) - 1;
energy_thresh_2 = 0.007;
energy_thresh_3 = 0.03;

denoised_data = zeros(1, length_indata);

han_window = hanning(window_length);

qmf = makeonfilter('Symmlet' , 7);


for wcounter = 0 : floor(length_indata / step_length) - 2
 wcounter
 wdata = indata(wcounter * step_length + 1  : wcounter * step_length + window_length ) .*
han_window;
 wc_wdata = fwt_po(wdata , L , qmf);
%*** Calculate the wavelet coefficients of the signal...
 for i = maxlevel : -1 : L
   eval(['wc_wdata_' int2str(i) ' = wc_wdata(dyad(' int2str(i) '));']);
 end
%*** Calculate the energies of the subsignals...
 for i = maxlevel : -1 : L
   eval(['energy_wc_wdata(' int2str(i) ') = sum( wc_wdata_' int2str(i) '.* wc_wdata_'
int2str(i) ');']);
 end

 energy_wdata = sum(wdata.*wdata);

 for i = maxlevel : -1 : L
   if ( energy_wc_wdata(i) / energy_wdata ) < energy_thresh_2
```

```
                        wc_wdata(dyad(i)) = zeros(1 , length(dyad(i)));
                        energy_wc_wdata(i) = 0;
   end
  end

 %*** Shaping the wavelet coefficients...
  for i = maxlevel : -1 : L
   if (energy_wc_wdata(i) / energy_wdata ) < energy_thresh_3
    if energy_wc_wdata(i) ~= 0
                        weight = zeros(1 , length(dyad(i)));
                        dummy_wc = zeros(window_length , 1);
                        dummy_wc(dyad(i)) = wc_wdata(dyad(i));
                        [dummy_wc , weight]=wc_shape(dummy_wc,wdata,i,qmf,1,0);
                        wc_wdata(dyad(i)) = dummy_wc(dyad(i));
    end
   end
  end

 denoised_wdata = iwt_po(wc_wdata , L , qmf)';

  if wcounter ==0
    denoised_data( 1 : window_length) = denoised_wdata;
  else
    denoised_data(wcounter * step_length + 1 : (wcounter + 1) * step_length) =
denoised_data(wcounter * step_length + 1 : (wcounter +1) * step_length) +
denoised_wdata(1 : step_length);
    denoised_data((wcounter + 1) * step_length + 1 : (wcounter + 2) * step_length) =
denoised_wdata(step_length + 1 : 2 * step_length);
  end

% main loop end!
end


fid = fopen('zmgmet2.raw', 'w');
fwrite(fid,denoised_data,'int16');
fclose(fid);
```

```
%*********************************************************************
%                    MS Thesis Work
% Name : wc_shape.m
% By: Emre Baris Aksu
% Creation Date : 10/3/1998
% Last Update   : 30/7/1998
%*********************************************************************
%
% Description:
%
%       This function shapes the wavelet coefficients according to the
%       energy of the original signal's windows corresponding to the
%       wavelet coefficients.
%                       wc_shape(wc, x , k , h0 , gain , a);
%
%               wcb             : The wavelet coefficients of the band
%               x               : Signal to be denoised
%               k               : The level of the coefficients
%               h0              : The qmf filter coefficients.
%               gain            : The normalized scale factor.
%               a               : A DC value added. ( between 0 and 1)
%*********************************************************************


function [shaped_wc , weight] = wc_shape(wcb ,x , k , h0 , gain , a);

wc_total = fwt_po(x , 1 , h0);
wc = wc_total( dyad( k ) );
delta = round( length(x) / length(dyad(k)) );
L=1;
x1 = iwt_po(wcb, L, h0);

for i = 0 : length( wc ) - 1,
  weight(i + 1) = sum( x1(i * delta + 1 : (i + 1) * delta) .* x1(i * delta + 1 : (i + 1) * delta) );
end;

maxw=max(weight);
weight = weight / maxw;

weight = weight + a;
maxw=max(weight);

weight = ( gain / maxw ) * weight ;
weight = sqrt(weight);
temp = wc .* weight';

shaped_wc = zeros(1 , length (x)) ;
shaped_wc( dyad( k )) = temp;
```

B.3 Method3.m

```
%*********************************************************************
%                       MS Thesis Source Code - 3
% Name : method3.m
% By: Emre Baris Aksu
% Creation Date : 21/6/1998
% Last Update   : 28/11/1998
%*********************************************************************
%
% Descripton:
%        This code is the implementation of Method-3. It uses Wavelab v7.00 library.


clear;
fid = fopen('zmg.raw' , 'r');
indata = fread(fid , 'int16');
fclose(fid);
fs = 22050;

length_indata = length(indata);
D = 3;
window_length = 1024;
step_length = window_length / 2;
maxlevel = log2(window_length) - 1;
entropy_threshold = 0.02;
maxrepeat = 2;

han_window = hanning(window_length);
qmf = makeonfilter('Daubechies' , 14);


denoised_sum = zeros(1, length_indata);

for repeat_process_counter = 1 : maxrepeat

denoised_data = zeros(1, length_indata);


for wcounter = 0 : floor(length_indata / step_length) - 2
 repeat_process_counter
 wcounter

 wdata = indata(wcounter * step_length + 1  : wcounter * step_length + window_length ) .*
han_window;

 wp_wdata = wpanalysis(wdata , D , qmf);
 stree = calcstattree(wp_wdata , 'Entropy');
 [btree , vtree] = bestbasis(stree , D);

 dummy_wp = zeros(window_length , D+1);
 node_magnitudes = zeros( 2 ^ (D+1) - 1 , 1);
 node_entropies_normalized = zeros( 2 ^ (D+1) - 1 , 1);

 bestbasis_coeffs = unpackbasiscoeff(btree , wp_wdata);
 dummy_wp = packbasiscoeff(btree , dummy_wp , bestbasis_coeffs);
```

```
 ss = norm(wp_wdata(:,1));
 for d = 0 : D
  for b = 0 : 2^d - 1
   node_magnitudes(node(d,b)) = sum(dummy_wp(packet(d,b,window_length),d+1) .*
dummy_wp(packet(d,b,window_length),d+1));
          p = (dummy_wp(packet(d,b,window_length),d+1)./ss).^2;
   e = - sum(sum(p .* log2(eps+p)));
          if node_magnitudes(node(d,b)) ==0
           node_entropies_normalized(node(d,b)) = 0;
   else
    node_entropies_normalized(node(d,b)) = (2^e) / (length(packet(d,b,window_length)));
   end
  end
 end

 synth_wp = zeros(window_length , D+1);

 for d = 0 : D
  for b = 0 : 2^d - 1
   if node_entropies_normalized(node(d,b)) == max(node_entropies_normalized)
    synth_wp(packet(d,b,window_length), d+1) = dummy_wp(packet(d,b,window_length),
d+1);
   end
  end
 end

 denoised_wdata = wpsynthesis(btree , synth_wp , qmf);


 if wcounter ==0
  denoised_data( 1 : window_length) = denoised_wdata;
 else
  denoised_data(wcounter * step_length + 1 : (wcounter + 1) * step_length) =
denoised_data(wcounter * step_length + 1 : (wcounter +1) * step_length) +
denoised_wdata(1 : step_length);
  denoised_data((wcounter + 1) * step_length + 1 : (wcounter + 2) * step_length) =
denoised_wdata(step_length + 1 : 2 * step_length);
 end

% main loop end!
end
 denoised_sum = denoised_sum + denoised_data ;
 indata = indata - denoised_data';

end

fid = fopen('d:\matlab42\dist\zmgmet3.raw', 'w');
fwrite(fid , denoised_sum , 'int16');
fclose(fid);
```