A JAVA TOOLBOX FOR WAVELET BASED IMAGE DENOISING

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

BY

GÜNEY TUNCER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
GEODETIC AND GEOGRAPHIC INFORMATION TECHNOLOGIES

DECEMBER 2006

Approval of the Graduate School of Natural and Applied Sciences.

_____
Prof. Dr. Canan Özgen
Director

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

_____
Assist. Prof. Dr. Zuhal Akyürek
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.

_____
Assoc. Prof. Dr. Mahmut Onur Karslıoğlu
Supervisor

Examining Committee Members

Prof. Dr. Vedat Toprak (METU,GEOE) _____

Assoc. Prof. Dr. Mahmut Onur Karslıoğlu (METU,CE) _____

Dr. Jürgen Friedrich (BAŞKENT ÜNV., CENG) _____

Erol Tunalı (TÜBİTAK-BİLTEN) _____

Assist. Prof. Dr. Bahattin Coşkun (METU,CE) _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Name, Last name :

Signature :

ABSTRACT

A JAVA TOOLBOX FOR WAVELET BASED IMAGE DENOISING

TUNCER, Güney
M.Sc., Department of Geodetic and Geographic Information Technologies
Supervisor: Assoc. Prof. Dr. Mahmut Onur KARSLIOĞLU

December 2006, 108 pages

Wavelet methods for image denoising have became widespread for the last decade. The effectiveness of this denoising scheme is influenced by many factors. Highlights can be listed as choosing of wavelet used, the threshold determination and transform level selection for thresholding. For threshold calculation one of the classical solutions is Wiener filter as a linear estimator. Another one is VisuShrink using global thresholding for nonlinear area. The purpose of this work is to develop a Java toolbox which is used to find best denoising schemes for distinct image types particularly Synthetic Aperture Radar (SAR) images. This can be accomplished by comparing these basic methods with well known data adaptive thresholding methods such as SureShrink, BayeShrink, Generalized Cross Validation and Hypothesis Testing. Some non-wavelet denoising process are also introduced. Along with simple mean and median filters, more statistically adaptive median, Lee, Kuan and Frost filtering techniques are also tested to assist wavelet based denoising scheme. All of these methods on the basis of wavelet models and some traditional methods will be

implemented in pure java code using plug-in concept of ImageJ which is a popular image processing tool written in Java.

Keywords: Wavelet transforms, Noise reduction, Statistical methods, SAR, Java

ÖZ

GÖRÜNTÜLERDEKİ GÜRÜLTÜLERİN TEMİZLENMESİ İÇİN
DALGACIK TABANLI JAVA TAKIM KUTUSU

TUNCER, Güney
Yüksek Lisans, Jeodezi ve Coğrafi Bilgi Sistemleri Bölümü
Tez Yöneticisi: Doç. Dr. Mahmut Onur KARSLIOĞLU

Aralık 2006, 108 sayfa

Görüntüler üzerinde gürültü temizleme için dalgacık yöntemlerinin kullanımı son zamanlarda oldukça yaygın bir hale gelmiştir. Dalgacık eşikleme (wavelet thresholding) olarak tabir edilebilecek bu işin başarısı pek çok değişkene bağlıdır. Bunların arasında uygun dalgacık modelinin seçimi, eşik hesaplama teknikleri ve eşik uygulama seviyesi en başta listelenebilir. Eşik hesabı için kullanılan "Wiener" filitresi doğrusal (linear) hesaplama tekniği kullanan klasik yöntemlerden biridir. Doğrusal olmayan (nonlinear) metotların en basit örneği ise evrensel eşik (global thresholding) mantığına dayalı VisuShrink'dir. Bu çalışmadaki amaç farklı görüntü tipleri için ve özellikle Synthetic Aperture Radar (SAR) görüntüleri için en başarılı gürültü temizleme kombinasyonlarını belirleyebilecek bir Java takım kutusu oluşturmaktır. Bu da bahsi geçen temel eşikleme hesapları ile gelişmiş istatiksel hesaplama teknikleri "SureShrink", "BayeShrink", "Generalized Cross Validation" ve "Hypothesis Test" kıyaslanmasıyla yapılacaktır. Bunların yanısıra çalışmaya yardımcı olmak amacıyla mean, medyan, uyarlamalı medyan, Lee, Kuan ve Frost gibi dalgacık

eşikleme yöntemi olmayan bazı hesaplamalara yer verilmektedir. Tüm bu dalgacık tabanlı hesaplama teknikleri ve bazı geleneksel yöntemler Java programlama dili kullanılarak popüler bir görüntü işleme uygulaması olan "ImageJ" altında geliştirilmektedir.

Anahtar kelimeler: Dalgacık dönüşümleri, Gürültü azaltma, İstatiksel yöntemler, SAR, Java

ACKNOWLEDGEMENTS

My gratitude is expressed to Assoc. Prof. Dr. Mahmut Onur Karslıoğlu for his guidance, support throughout my research and numerous suggestions he made to this thesis.

I am very grateful to Banu, Güler, Sehlan Tuncer, my sister, my mother and my father, for supporting me throughout my educational life.

I also thank Mahmut Arıkan, Ramon Hanssen, and Nuretdin Kaymakçı for providing the SAR image.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES

LIST OF TABLES

TABLES

LIST OF SYMBOLS

x_d	the horizontal line of image
y_d	the vertical line of image
x	noisy data set
xr	reversed noisy data set
x_i	specific data in noisy data set
xr_i	reversed specific data in noisy data set
X	noisy data parent information
y	noise free data set
x_s	simulated data set
Ψ	wavelet function
φ	scaling function
j	scale number
$\gamma_{j,k}$	True wavelet coefficients
$w_{j,k}^{(n)}$	Emprical wavelet coefficents
$d_{j,k}$	wavelet details of noisy data
G_o	low pass filter
H_o	high pass filter
$d()$	detail information produced by H_o
$a()$	coarse approximations *produced by* G_o

G_{11}	synthesis filter
N	length of data
M	number of pixels in the image
T_c	threshold value
T_m	threshold multiplier
σ_i	user input standard deviation
σ_w	standard deviation of kernel window
σ_o	standard deviation of the noise in noisy image
σ_s	standard deviation of wavelet coefficients of simulated image
med_w	median value of kernel window
ε	unbiased estimator of the mean square errors
E{.}	expectation operator
H_o	null hypothesis
H_a	alternative hypothesis
p	probability
Γ()	gamma distribution function
$F()$	Chi-square distribution function
α	significance value for hypothesis test
$m()$	mean integrated square error function
STD	Standard Deviation
MSE	Mean Square Error
MISE	Mean Integrated Squared Error
PSNR	Peak Signal to Noise Ratio

MAX_I	Maximum Pixel Value of The Image
B	color dept "Bit" of an image
MAD	Median Absolute Deviation
SURE	Stein Unbiased Risk Estimator
BAMS	Bayesian Adaptive Multiresolution Shrinker.
med_w	median value of kernel

SUBSCRIPTS

WT	Wavelet Transformation
WC	Wavelet Coefficients
DWT	Discrete Wavelet Transform
STFT	Short Time Fourier Transform
CDF	Cohen Daubechies Feauveau
GCV	Generalized Cross-Validation
DF	Double Filtering
WDT	Wavelet Denoising Toolbox
JRE	Java Runtime Environment
JDK	Java Development Kit
FIR	Finite Impulse Response

CHAPTER 1

INTRODUCTION

Denoising is one of the most important part of image processing. We are dealing with data which contain noise almost every time. Many scientific works have been done in this area. Wavelet theory relatively is the newest one though its mathematical underpinnings date back to the Fourier transformation. The main algorithm of wavelet theory dates back to the work of Stephane Mallat in [2, 5]. Since then, research on wavelets has spreaded out. Famous contributors in this decade can be listed as Ingrid Daubechies, Ronald Coifman, and Victor Wickerhauser [3, 6] .

Denoising operations in wavelet transform domain (WTD) is the second part of wavelet denoising scheme. Donoho and Johnstone who proposed hard and soft thresholding schemes on WTD are well kown contributors [19]. On the basis of hard and soft thresholding schemes many advanced non linear statistical methods were applied to the WTD by Donoho and Jonhstone, Nason G.P., Odgen and Parzen [20, 26, 23, 24]. Well known non linear Bayesian estimator were applied to the WTD by Vidakovic and Ruggeri [30]. In linear denoising case Wiener filtering was also introduced in WTD [18]. Later some of these statistical methods were modified to compute a separate threshold for each wavelet resolution level.

In this work main focus is to develop a versatile wavelet denoising toolbox as bringing up popular wavelet models and denoising algorithms. Decimated wavelet transformation will be done using orthogonal and biorthogonal wavelet models. Denoising operations in wavelet domain will be studied in two groups as linear and non-linear. In the case of non-linear algorithms most popular data adaptive

methods such as SureShrink, BayesShrink, Generalized Cross Validation and Hypothesis testing will be applied along with VisuShrink as non-data adaptive method. On the other hand Wiener filtering will be employed in terms of linear algorithm. Non-wavelet filters such as traditional Mean, Median, and more statistically Adaptive Median, Lee, Kuan and Frost filters will be used also in this work. As a result using wavelet based filters with the aid of non-wavelet filters, the most appropriate denoising combinations are found out for different image types such as astronomical images, satellite images, magnetic resonance images (MRI), synthetic aperture radar (SAR) images or other ordinary images by using the toolbox generated. Implemented wavelet models will also be tested with best denoising combinations for SAR images.

In Chapter 2, image handling is discussed. ImageJ one of the most powerful image editing tool written in Java is being used for image handling and displaying. It is still under development and free licenced. There are thousands of plugins written for ImageJ but no significant work available in wavelet theory in context with sophisticated denoising algorithm. This work will also be the most complete wavelet based denoising plugin for ImageJ. In chapter 3, wavelet transformation will be explained and popular wavelet models will be compared. In Chapter 4 wavelet based denoising algorithms will be presented and results of these filters will be discussed. In Chapter 5, non-wavelet denoising filters will be applied. The new Toolbox generated throughout this work will be presented in Chapter 6. Finally, in Chapter 7 best combinations of wavelet models, wavelet based filters and non-wavelet filters will be found out for specific images.

CHAPTER 2

IMAGE HANDLING AND DISPLAYING

Wavelet based image denoising is based on obtaining pixel values of images as proper data sets. Ordinary images seen on the computer screen have two dimensions, in terms of x_d and y_d representing data in the horizontal and vertical axes. A pixel called picture element is known as one of the many tiny dots that make up the representation of a picture in a screen of a computer. Each dot may have different values. With these explanations one can say that image with "*256x512"* resolution have "*256"* dots in x_d and "*512"* dots in y_d.

Wavelet transformation (WT) which will be treated in Chapter 3, was initially designed on linear "one dimensional" signal, however this method can be extended to signals for higher order dimensions successfully. The base of multi-dimensional WT is completely same with the one-dimensional WT. One dimensional WT is applied to each dimension respectively.

$$x = x_d \tag{2.1}$$

$$x = y_d \tag{2.2}$$

where x_d, and y_d are noisy data set and x represents the dataset used for WT. In (2.1) x takes values of x_d. In (2.2) x takes again y_d values to make same process after WT on x_d.

Acquiring values of images in both dimensions can be done easily with integrated image processor libraries of ImageJ which will be discussed in the next sections. Displaying the image on the screen is also done with the same integrated libraries.

2.1 General Properties of ImageJ

ImageJ is one of the world's fastest pure Java image processing program. It runs on most popular operating systems such as Linux, Mac OS 9, Mac OS X, Windows. ImageJ is written in Java so it needs a Java Runtime Environment (JRE) or Java Development Kit (JDK) to operate. ImageJ and its java source code are freely available and in the public domain [31]. No license is required. It can read, display, edit, analyze, process, save and print many of the well known images up to 32-bit color quality. It is multithreaded, so time-consuming operations such as image file reading can be performed in parallel with other operations.

Custom acquisition, analysis and processing plugins can be developed using ImageJ's built-in editor and Java compiler. User written plugins make it possible to solve almost any image processing or analysis problem.

2.2 Plug-in Concept of ImageJ

The functions provided by ImageJ's menu commands (most of them) can be extended by user plugins. These plugins are Java classes implementing the necessary placed in a certain folder. They can be conveniently compiled inside ImageJ or Windows environment. Plugins found by ImageJ are placed in the plugins menu.

2.3 Methods and Classes of ImageJ

ImageJ includes several classes and methods to operate on image. Mostly used methods called *ImageProcessor* and *ImagePlus* contain all necessary objects for

creation of a new image and accessing or editing purposes. *ImageWindow* class can be used to display the image.

2.3.1 Creating New Images

In many cases it will make sense that a plugin does not modify the original image, but creates a new image that contains the modifications. ImagePlus's *createImagePlus()* method returns a new ImagePlus with this ImagePlus's attributes, but no image. A similar function is provided by ImageProcessor's *createProcessor()* method which returns a new, blank processor with specified width and height which can be used to create a new ImagePlus using the constructor *ImagePlus()*. The class NewImage offers some useful static methods for creating a new ImagePlus of a certain type. ImagePlus's *createByteImage()* method creates a new 8-bit grayscale or color image with the specified title, width and height and number of slices.

2.3.2 Accessing and Editing Pixel Values of Noisy Image

Retrieving the pixel values of noisy image can be done by using an ImageProcessor's *GetPixels()* object. It returns a reference to this image's pixel array. As the type of this array depends on the image type we need to cast this array to the appropriate type when we get it.

To convert a position in this array to a (x,y) coordinate in an image, we need at least the width of a scanline. The width and height of an ImageProcessor can be retrieved using these methods: *getHeight()* and *getWidth()* object.

ImageJ have two processor reading pixel values. These are *ByteProcessors* and *ColorProcessors*. In this work greyscale images which have values from "0" to

"255" will be used. Java's *byte* data type used in *ByteProcessor* is perfect for grey scale images but has values ranging from "–128" to "127". Sign bit must be eliminated. This can be done using a binary *AND*.

Other mostly used objects can be listed as fallows : *getPixel()* returns the value of the specified pixel. *putPixels()* sets the pixel at (x_d, y_d) to the specified value. *getColumn()* returns the pixels down the column starting at (x_d, y_d). *putColumn()* inserts the pixels contained in data into a column starting at (x_d, y_d). *getRow()* returns the pixels along the horizontal line starting at (x_d, y_d). *putRow()* inserts the pixels contained in data into a horizontal line starting at (x_d, y_d). *getLine()* returns the pixels along the line (x_{d_1}, y_{d_1}) / (x_{d_2}, y_{d_2}).

2.3.3 Displaying Images

ImageJ uses a class called ImageWindow to display ImagePlus images. ImagePlus contains everything that is necessary for updating or showing newly created images. Procedures in this class such as *draw()* displays this image, *updateAndDraw()* updates this image from the pixel data in its associated ImageProcessor, and displays it, updateAndRepaintWindow() calls *updateAndDraw()* to update from the pixel data and draw the image, and also repaints the image window to force the information displayed above the image (dimension, type, size) to be updated. *Show()* opens a window to display this image and clears the status bar, *hide()* closes the window, if any, that is displaying this image.

2.4 Implementation Tips

In this work input image was limited to 8-bit pixel value. Images with higher colors of dept such as 16-bit and 32 bit can be handled with ImageJ but this will cause some performance lacks due to complex processing.

Because of the wavelet filter's nature, discreet wavelet transform mentioned in chapter 3 can be done on proper dataset. Length of dataset in both dimension x_d and y_d must be equal and power of 2 such as "128x128", "512x512" or "2048x2048". Any other non power of two arbitrary length data set is converted to suitable length before the transform begin. Assume that there is an image with "345x850" resolution. After the conversion new image have "1024x1024" resolution. Processed and reconstructed new image is again converted to original size at the end.

CHAPTER 3

WAVELET TRANSFORM

Wavelet theory has been one of the most useful development in the last decade that developed independently on several fronts. Different signal and image processing techniques had significant contributions in this theory [1]. Some of the major contributors to this area can be listed as: multi-resolution signal processing, wavelet series expansion in applied mathematics, sub-band coding used in image and voice compression [2, 3, 4].

The reason of the most wavelet research is to build more efficient wavelet function which gives precise description of the signal. It is very complicated process to develop best wavelet function. But on the basis of several characteristics of the wavelets, the most suitable one can be determined for a given application.

In this chapter, the wavelet transformation is introduced, wavelet models are classified, multiresolution wavelet transformation on images is examined using the computational point of view. Features of different wavelet transform filters are examined on different image types.

3.1 Introduction to Wavelet Transform

A wavelet is a small wave with finite energy, which has its energy concentrated in time or space area to give ability for the analysis of time-varying phenomenon in other words it provides a time-frequency representation of the signal. Comparison of a wave with a wavelet is shown in Figure 3-1. Left graph is a

Sine Wave with infinite energy and the right graph is a Wavelet with finite energy.

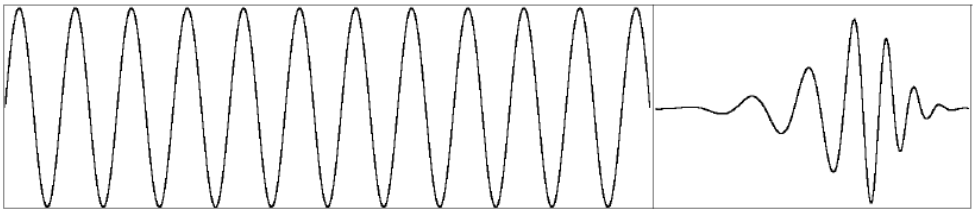


Figure 3-1 Comparison of a wave and a wavelet

Wavelet transformation (WT) was developed to overcome the shortcoming of the Short Time Fourier Transform (STFT), which can also be used to analyze non-stationary signals. While STFT gives a constant resolution at all frequencies, WT uses multi-resolution technique for non-stationary signals by which different frequencies can be analyzed with different resolutions [5].

Wavelet theory is based on analyzing signals to their components by using a set of basis functions. One important characteristic of the wavelet basis functions is that they relate to each other by simple scaling and translation. The original wavelet function is used to generate all basis functions. It is generally designed with respect to desired characteristics of the associated function. For multiresolution transformation, there is also a need for another function which is known as scaling function. It makes analysis of the function to finite number of components.

WT is a two-parameter expansion of a signal in terms of a particular wavelet basis functions or wavelets. Continuous wavelet transform (CWT) is written as:

$$\gamma(s,\tau) = \int f(t)\psi_{s,\tau}(t)dt \tag{3.1}$$

This equation shows how a function $f(t)$ is decomposed into a set of basis functions $\psi_{s,\tau}(t)$, called the wavelets. The variables s and τ, scale and translation, are the new dimensions after the wavelet transform.

Inverse wavelet transformation can be expressed as :

$$f(t) = \iint \gamma(s,\tau)\psi_{s,\tau}(t)d\tau ds \tag{3.2}$$

The wavelets are generated from a single basic wavelet $\psi(t)$, the so-called *mother wavelet*, by scaling and translation:

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right) \tag{3.3}$$

For different integer values of s and τ, integer τ represents translation of the wavelet function and represents time or space in WT. Integer s, however, is an indication of the wavelet frequency or spectrum shift and generally referred to as scale. For demonstration purposes, two different scaled versions of a wavelet along with the mother wavelet are shown in Figure 3-2.

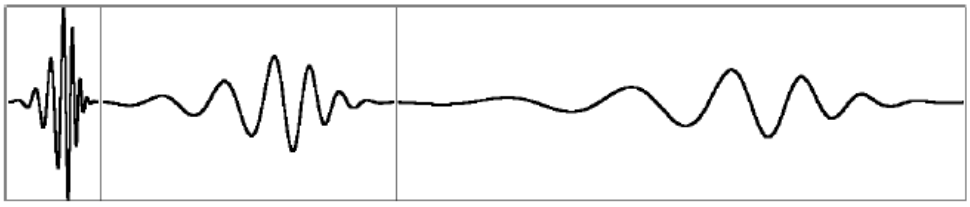


Figure 3-2 Mother wavelet and its two different scaled versions

In Figure 3-2 the left graph is the mother wavelet ψ_{D12}, the middle one is the wavelet at scale $s = -1$ and the right one is the wavelet at scale $s = -2$. The other way to look at these graphs is: to assume that the right graph is the mother wavelet, the middle one is the wavelet at scale $s = 1$ and the left one is the wavelet at scale $s = 2$.

It is easily seen that different scales referred to different frequency spectrum. If we only look at the center frequency of each spectrum, it is seen that the center frequency changes by a factor of two for each increment or decrement of integer scale *s*. For simplicity, in wavelet transform reference to frequency is replaced by reference to scale. Higher scale corresponds to finer localization and vice versa [8].

3.2 Wavelet Properties

There are two important properties of wavelets. These are the admissibility and the regularity conditions. It can be shown that square integrable functions $\psi(t)$ satisfying the *admissibility condition* [7].

$$\int \frac{|\psi(w)|^2}{|w|} dw < +\infty \tag{3.4}$$

This can be used to first analyze and then reconstruct a signal without loss of information. In (3.4) $\psi(w)$ stands for the Fourier transform of $\psi(t)$. The admissibility condition implies that the Fourier transform of $\psi(t)$ vanishes at the zero frequency, i.e.

$$|\psi(w)|^2\Big|_{w=0} = 0 \tag{3.5}$$

This means that wavelets must have a band-pass like spectrum. A zero at the zero frequency also means that the average value of the wavelet in the time domain must be zero,

$$\int \psi(t) dt = 0 \tag{3.6}$$

and therefore it must be oscillatory. In other words, $\Psi(t)$ must be a *wave*. According to (3.1) the WT of a one-dimensional function is two-dimensional. The time-bandwidth product of the wavelet transform is the square of the input signal. Some additional conditions are imposed on the wavelet functions in order to make the WT decrease quickly with decreasing scale s. These are the *regularity conditions* and they state that the wavelet function should have some smoothness and concentration in both time and frequency domains [10].

WT in (3.1) can be expanded into the Taylor series at $t = 0$ until order n (let $\tau = 0$ for simplicity) [7]:

$$\gamma(s,0) = \frac{1}{\sqrt{s}}\left[\sum_{p=0}^{n} f^{(p)}(0)\int \frac{t^p}{p!}\psi\left(\frac{t}{s}\right)dt + O(n+1)\right] \tag{3.7}$$

Here $f^{(p)}$ stands for the p^{th} derivative of f and $O(n+1)$ means the rest of the expansion. Now, if *moments* of the wavelet is defined by M_p,

$$M_p = \int t^p \psi(t)dt \tag{3.8}$$

then equation (3.7) can be rewritten as :

$$\gamma(s,0) = \tag{3.9}$$
$$\frac{1}{\sqrt{s}}\left[f(0)M_0 s + \frac{f^{(1)}(0)}{1!}M_1 s^2 + \frac{f^{(2)}(0)}{2!}M_2 s^3 + \ldots + \frac{f^{(n)}(0)}{n!}M_n s^{n+1} + O(s^{n+2})\right]$$

Admissibility condition states that at 0^{th} moment $M_0 = 0$ so that the first term in the right-hand side of (3.9) is zero. If the other moments are maked as zero up to M_n zero as well, then the wavelet transform coefficients $\Upsilon(s,\tau)$ will decay as fast as s^{n+2} for a smooth signal $f(t)$. This is known in literature as the *vanishing moments* or approximation order [33]. If a wavelet has N vanishing moments,

then the approximation order of the wavelet transform is also *N*. The moments do not have to be exactly zero, a small value is often good enough. In fact, experimental research suggests that the number of vanishing moments required depends heavily on the application [34].

3.3 Discrete Wavelets

As mentioned before the CWT maps a one-dimensional signal to a two-dimensional time-scale joint representation that is highly redundant. The time-bandwidth product of the CWT is the square of that of the signal and for most applications, which seek a signal description with as few components as possible, this is not efficient. To overcome this problem *discrete wavelets* have been introduced. Discrete wavelets are not continuously scalable and translatable but can only be scaled and translated in discrete steps. This is achieved by modifying the wavelet representation in (3.3) to create following equaiton [6].

$$\psi_{j,k}(\mathrm{t}) = \frac{1}{\sqrt{s_0^j}}\psi\left(\frac{t - k\tau_0 s_0^j}{s_0^j}\right) \tag{3.10}$$

Although it is called a discrete wavelet, it normally is a piecewise continuous function. In (3.10) j and k are integers and $s_0 > 1$ is a fixed dilation step. The translation factor τ_0 depends on the dilation step. The effect of discretizing the wavelet is that the time-scale space is now sampled at discrete intervals. Value of s_0 is usually equals to "2" so that the sampling of the frequency axis corresponds to *dyadic sampling*. This is a very natural choice for computers, as well as the human ear and music for instance. τ_0 which is known as the translation factor takes value "1" in order to achieve dyadic sampling of the time axis [10].

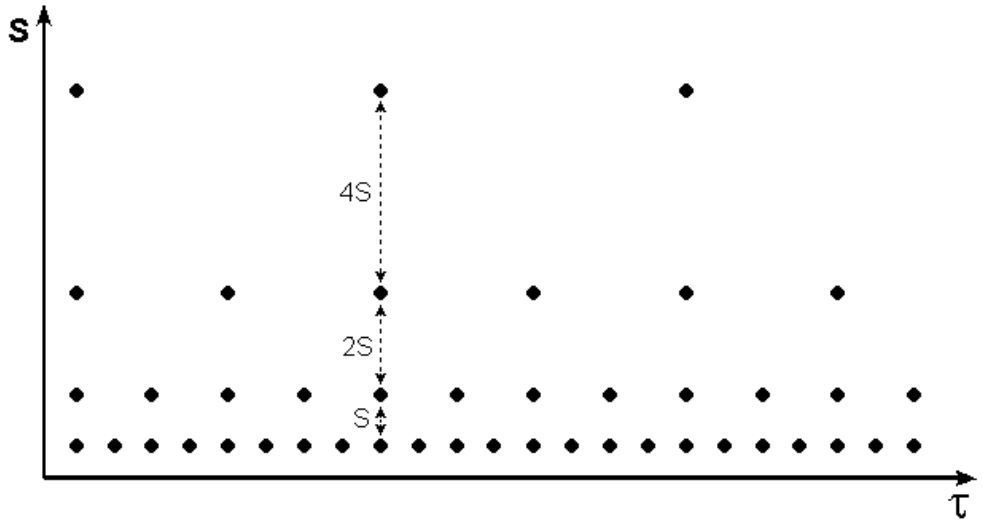


Figure 3-3 Localization of the discrete wavelets in the time-scale space on a dyadic grid [10].

When discrete wavelets are used to transform a continuous signal the result will be a series of wavelet coefficients, and it is referred to as the *wavelet series decomposition*. An important issue in such a decomposition scheme is of course about the reconstruction. It is all very well to sample the time-scale joint representation on a dyadic grid, but if it will not be possible to reconstruct the signal it will not be of great use. As it turns out, it is indeed possible to reconstruct a signal from its wavelet series decomposition. It is proven that the necessary and sufficient condition for stable reconstruction is that the energy of the wavelet coefficients must lie between two positive bounds [6].

$$A\|f\|^2 \le \sum_{j,k} \left|\langle f, \psi_{j,k} \rangle\right|^2 \le B\|f\|^2 \tag{3.11}$$

where $\| f \|^2$ is the energy of $f(t)$, $A > 0$, $B < \infty$ and A, B are independent of $f(t)$. When expression (3.11) is satisfied, the family of basis functions $\psi_{j,k}(t)$ with j, $k \in Z$ is referred to as a *frame* with frame bounds A and B. When $A = B$ the frame is tight and the discrete wavelets behave exactly like an orthonormal basis. When $A \neq B$ exact reconstruction is still possible at the expense of a *dual frame*.

In a dual frame discrete wavelet transform the decomposition wavelet is different from the reconstruction wavelet.

The discrete wavelets can be made orthogonal to their own dilations and translations by special choices of the mother wavelet, which means:

$$\int \psi_{j,k}(t)\psi^{*}_{m,n}(t)dt = \begin{cases} 1 & if\ j = m\ and\ k = n \\ 0 & otherwise \end{cases} \tag{3.12}$$

An arbitrary signal can be reconstructed by summing the orthogonal wavelet basis functions, weighted by the wavelet transform coefficients [33]:

$$f(t) = \sum_{j,k} \gamma(j,k)\psi_{j,k}(t) \tag{3.13}$$

Equation (3.13) shows the inverse wavelet transform for discrete wavelets.

3.4 The Scaling Function

The question is "*How to cover the spectrum all the way down to zero ?*" [10]. Because every time the wavelet is streched in the time domain with a factor of 2, its bandwidth is halved. In other words, with every wavelet stretch only half of the remaining spectrum is covered, which means that infinite number of wavelets will be needed to get the job done.

The solution to this problem is simply not to try to cover the spectrum all the way down to zero with wavelet spectra, but to use a cork to plug the hole when it is small enough. This cork then is a low-pass spectrum and it belongs to the so-called scaling function [10]. The scaling function was introduced by Mallat [2].

Because of the low-pass nature of the scaling function spectrum it is sometimes referred to as the *averaging filter*.

If we look at the scaling function as being just a signal with a low-pass spectrum, then we can decompose it in wavelet components and express it like in (3.13) .

$$\varphi(t)=\sum_{j,k}\gamma(j,k)\psi_{j,k}(t) \tag{3.16}$$

Since the scaling function $\varphi(t)$ is selected in such a way that its spectrum neatly fitted in the space left open by the wavelets, the expression (3.16) uses an infinite number of wavelets up to a certain scale j seen in Figure 3.4. This means that if a signal is analysed using the combination of scaling function and wavelets, the scaling function by itself takes care of the spectrum otherwise covered by all the wavelets up to scale j, while the rest is done by the wavelets. In this way the number of wavelets are limited from an infinite number to a finite number.

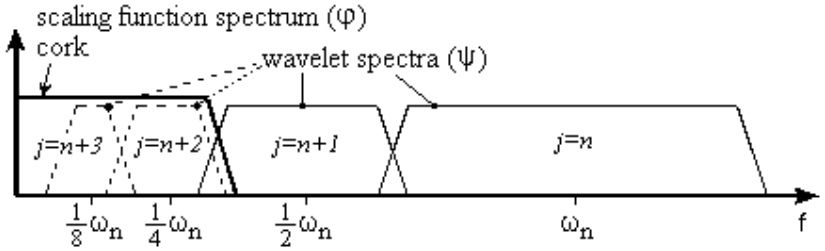


Figure 3-4 How an infinite set of wavelets is replaced by one scaling function [10].

Because of the low-pass spectrum of the scaling function admissibility condition can be expressed similar to (3.6)

$$\int \varphi(t)dt = 1 \tag{3.17}$$

which shows that the 0^{th} moment of the scaling function can not vanish [10].

3.5 Discrete Wavelet Transform

In (3.16) the scaling function was expressed in wavelets from minus infinity up to a certain scale *j*. If wavelet spectrum is added to the scaling function spectrum, this will give a new scaling function, with a spectrum twice as wide as the first. The effect of this addition is that first scaling function can be easily expressed in terms of the second, because all the information needed to do this is contained in the second scaling function. It can be expressed formally in the so-called multiresolution formulation [33].

$$\varphi(2^j t) = \sum_k h_{j+1}(k)\varphi(2^{j+1}t - k) \tag{3.18}$$

The two-scale relation states that the scaling function at a certain scale can be expressed in terms of translated scaling functions at the next smaller scale.

The first scaling function replaced a set of wavelets and therefore the wavelets can be expressed in this set in terms of translated scaling functions at the next scale. More specifically the wavelet can be written at level *j*:

$$\psi(2^j t) = \sum_k g_{j+1}(k)\varphi(2^{j+1}t - k) \tag{3.19}$$

which is the two-scale relation between the scaling function and the wavelet.

Since signal $f(t)$ could be expressed in terms of dilated and translated wavelets up to a scale j-1, this leads to the result that $f(t)$ can also be expressed in terms of dilated and translated scaling functions at a scale j :

$$f(t)=\sum_k \lambda_j(k)\varphi(2^j t-k) \tag{3.20}$$

For the next scale j-1, wavelets have to be added in order to keep the same level of detail. As a result signal $f(t)$ can be expressed as :

$$f(t)=\sum_k \lambda_{j-1}(k)\varphi(2^{j-1} t-k)+\sum_k \gamma_{j-1}(k)\psi(2^{j-1} t-k) \tag{3.21}$$

If the scaling function $\varphi_{j,k}(t)$ and the wavelets $\psi_{j,k}(t)$ are orthonormal or a tight frame, then the coefficients $\lambda_{j\text{-}1}(k)$ and $\gamma_{j\text{-}1}(k)$ are found by taking the inner products :

$$\begin{aligned}\lambda_{j\text{-}1}(k) &= \left\langle f(t),\varphi_{j,k}(t)\right\rangle \\ \gamma_{j\text{-}1}(k) &= \left\langle f(t),\psi_{j,k}(t)\right\rangle\end{aligned} \tag{3.22}$$

If $\varphi_{j,k}(t)$ and $\psi_{j,k}(t)$ in the inner products are replaced with suitably scaled and translated versions of (3.18) and (3.19) :

$$\lambda_{j-1}(k)=\sum_m h(m-2k)\lambda_j(m) \tag{3.23}$$

$$\gamma_{j-1}(k)=\sum_m g(m-2k)\gamma_j(m) \tag{3.24}$$

These two equations state that the wavelet and scaling function coefficients on a certain scale can be found by calculating a weighted sum of the scaling function coefficients from the previous scale.

Discrete weighted sum like the ones in (3.23) and (3.24) is the same as a digital filter and since the coefficients $\lambda_j(k)$ come from the low-pass part of the splitted signal spectrum, the weighting factors $h(k)$ in (3.23) must form a low-pass filter. And since the coefficients $\gamma_j(k)$ come from the high-pass part of the splitted signal spectrum, the weighting factors $g(k)$ in (3.24) must form a high-pass filter. This means that (3.23) and (3.24) together form one stage of an iterated digital filter bank [10]. The coefficients $h(k)$ is called *scaling filter* and the coefficients $g(k)$ is called as *wavelet filter*.

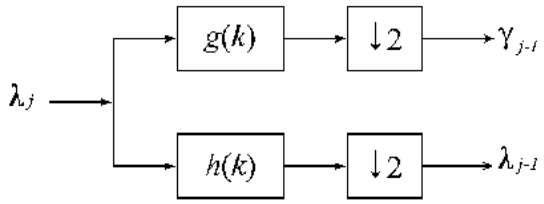


Figure 3-*5* One stage of an iterated filter bank.

3.5.1 Mallat Algorithm

The DWT is computed by successive low-pass and high-pass filtering of the discrete time-domain signal as shown in Figure 3-6. This is called the Mallat algorithm or Mallat-tree decomposition [2]. The low pass filter is denoted by G_o while the high pass filter is denoted by H_o. At each level, the high pass filter produces detail information, *d[n]*, while the low pass filter associated with scaling function produces coarse approximations, *a[n]*. The half band low pass filtering removes half of the frequencies and thus halves the resolution, the decimation by two doubles the scale.

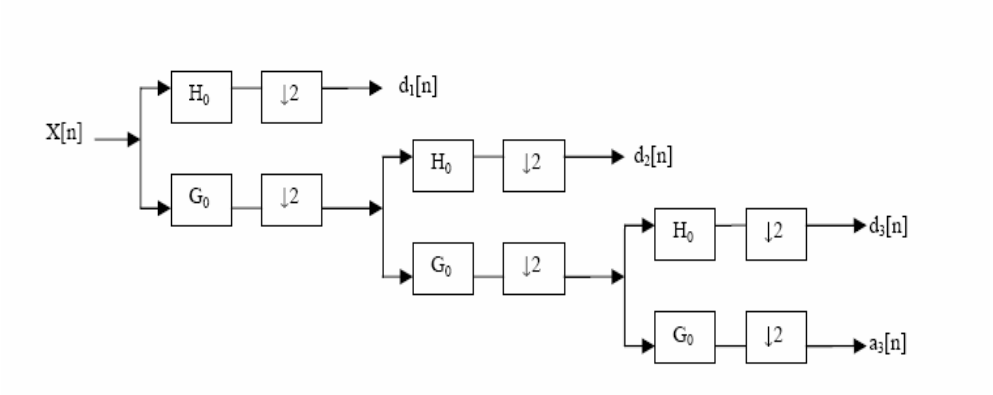


Figure 3-6 Mallat-tree decomposition

The filtering and decimation process is continued until the desired level is reached. The maximum number of levels depends on the length of the signal. This method is called decimated DWT in literature. There is also another method called undecimated DWT which has no decimate factor in scale calculation. In undecimated case, a signal is represented with the same number of wavelet coefficients at each scale. This means higher scales include coefficients of lower scales also. It is known that the use of non-decimated transforms minimizes the artifacts in the denoised data [18]. However decimated DWT gives more memory efficient performance with respect to undecimated one.

The DWT of the original signal is then obtained by concatenating all the coefficients, a[n] and d[n], starting from the last level of decomposition.

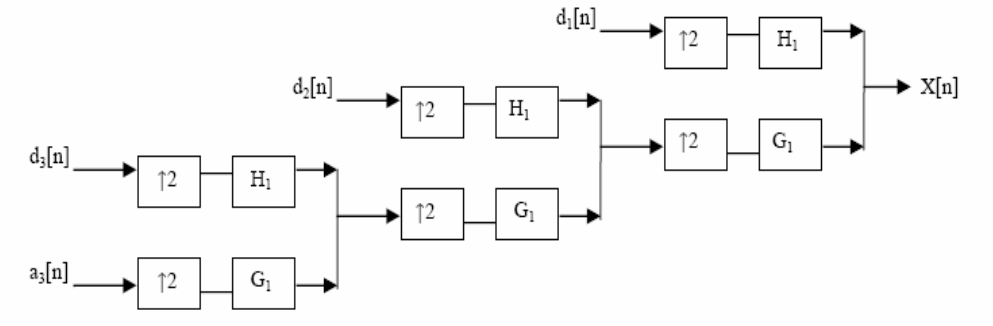


Figure 3-7 Mallat-tree reconstruction

Figure 3-7 shows the Mallat-tree reconstruction of the original signal from the wavelet coefficients [2]. Basically, the reconstruction is the reverse process of decomposition. The approximation and detail coefficients at every level are up-sampled by two, passed through the low pass and high pass synthesis filters and then added. This process is continued through the same number of levels as in the decomposition process to obtain and the original signal. The Mallat algorithm works equally well if the analysis filters, G_0 and H_0, are exchanged with the synthesis filters, G_{11}.

3.6 Classification of Wavelets

Wavelets can be divided in different classes in many different ways. For example, they can be divided based on their duration: infinite support wavelets and finite duration wavelets [8]. There are several infinite support wavelets such as Gaussian wavelets, Mexican Hat, Morlet, and Meyer. In this work infinite support wavelets will not be implemented. In practice, finite support (compact) wavelets are more popular due to their relations to multiresolution filter banks [8]. The most commonly used wavelets can be categorized into two classes: orthogonal and biorthogonal [5]. In orthogonal case the analysis and synthesis filters are not symmetric but this situation might be required in some applications. Biorthogonal wavelets are more complicated and are defined based on a pair of scaling and wavelet functions. In this case, the analysis and synthesis filters can be forced to be symmetric.

3.6.1 Features of Orthogonal Wavelets

Orthogonal wavelets have regular structure which leads to easy implementation and scalable architecture [5]. For orthogonal wavelet systems with real functions, the following conditions should be satisfied.

$$\psi_{j,k}(t).\ \psi_{m,n}(t)dt = 1 \text{ (if } j=m \text{ and } k=n\text{), 0 (Otherwise)}$$
$$\varphi_{j,k}(t).\ \varphi_{m,n}(t)dt = 1 \text{ (if } j=m \text{ and } k=n\text{), 0 (Otherwise)} \qquad (3.25)$$
$$\varphi_{j,k}(t).\ \psi_{j0}(t)dt = 0$$

Orthogonality property is an important property in any signal analysis operation. Remember that DWT decomposes a signal into bands that vary in spatial frequency and orientation (see section 3.5). Uniform quantization of a single band of coefficients results in an artifact that is the sum of a lattice of random amplitude basis functions of the corresponding DWT's wavelet function, which is called DWT uniform quantization error. The orthogonality property is required for the quantization error in the wavelet domain to be an exact indicator of the reconstructed image's final distortion [15].

Many functions exist that can satisfy the orthogonality requirements. Some of these functions are extraordinarily irregular, even fractal in nature. This may be an advantage in analyzing rough or fractal signals but it is likely to be a disadvantage for some signals and images. Orthogonal filters offer a high number of vanishing moments. This property is useful in many signal and image processing applications [8]. It has been shown that the number of vanishing moments of the wavelet, $\psi(t)$, is related to the smoothness or differentiability of $\varphi(t)$ and $\psi(t)$. The representation and approximation of polynomials, which are often a good model for certain signals and images, are also related to the number of vanishing or minimized wavelet moments. On the other hand, the number of zero moments in the scaling function $\varphi(t)$ is related to the goodness of the approximation of high resolution scaling coefficients by samples of the signal. This number also affects the symmetry and concentration of the scaling functions and wavelets [8].

3.6.2 Features of Biorthogonal Wavelets

Biorthogonal wavelets are known as two-band wavelets. They mean two different scaling functions $\varphi, \tilde{\varphi}$ which may generate different multiresolution analyses, and two different wavelet functions $\psi, \tilde{\psi}$. These functions provide the following conditions seen in (3.26) [12].

$$
\begin{aligned}
&< \varphi, \tilde{\varphi} > = 0, \\
&< \psi, \tilde{\psi} > = 0, \\
&< \psi, \tilde{\varphi} > \; = < \varphi, \tilde{\psi} > = 0,
\end{aligned}
\tag{3.26}
$$

where ψ, φ are the wavelet and scaling function $\tilde{\psi}, \tilde{\varphi}$ are another functions called dual wavelet and dual scaling function respectively.

In the case of the biorthogonal wavelets, wavelets and scaling functions do not have the same length. The scaling functions are always symmetric, while the wavelet function could be either symmetric or anti-symmetric.

For perfect reconstruction, biorthogonal filter bank has all odd length or all even length filters. The two analysis filters can be symmetric with odd length or one symmetric and the other antisymmetric with even length [9].

3.7 Wavelet Families

There are a number of bases functions that can be used as the mother wavelet for WT. Since the mother wavelet produces all wavelet functions used in the transformation through translation and scaling, it determines the characteristics of the resulting wavelet transform. Therefore, the details of the particular application

should be taken into account and the appropriate mother wavelet should be chosen in order to use the WT effectively.

Figure 3-8 illustrates some of the commonly used wavelet functions. Haar (a), Daubechies-4 (b), and Coiflets-1 (c) are finite orthogonal wavelets (see sections 3.7.1, 3.7.2 and 3.7.3) . Haar wavelet is one of the oldest and simplest wavelet [8]. Therefore, any discussion of wavelets starts with the Haar wavelet. These wavelets are capable of perfect reconstruction and they are chosen based on their shape and their ability to analyze the signal in a particular application [8].

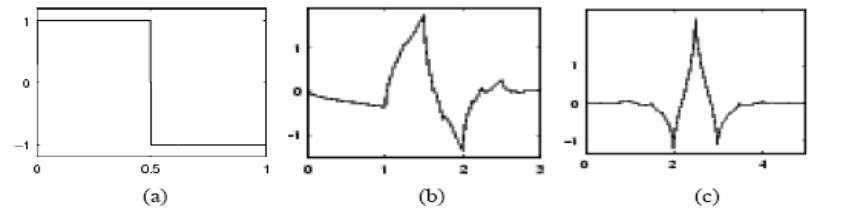


Figure 3-8 Various wavelet functions

3.7.1 Haar Wavelets

The Haar wavelet is the simplest and oldest one. It is a step function taking values 1 and -1. Its scaling functions can be expressed as following:

$$\varphi(t) = 1 \quad : \ 0 \le t < 1$$
$$\varphi(t) = 0 \quad : \ \text{otherwise} \qquad (3.28)$$
$$\Psi(t) = 1 \quad : \ 0 \le t < \tfrac{1}{2}$$
$$\Psi(t) = -1 \quad : \ \tfrac{1}{2} \le t < 1 \qquad (3.29)$$

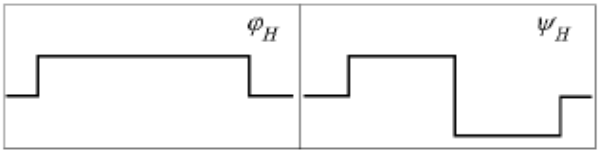


Figure 3-9 Haar scaling function (left) and Haar mother wavelet.

Haar wavelet is the only orthogonal wavelet that has symmetric analysis and synthesis filters. This particular wavelet has been studied extensively in the image processing area as Haar transform [8]. Graphs of Haar scaling function on the left and mother wavelet are shown in Figure 3-9.

3.7.2 Daubechies Wavelets

For a given wavelet order, Daubechies developed wavelets with maximum regularity [6]. In this case, the number of zero moments for $\psi(t)$ is maximized. Daubechies wavelets have the property of orthogonality, that is:

$$\begin{aligned} &< \varphi_i, \varphi_j > = 0, \text{for } i \neq j, \\ &< \Psi_i, \Psi_j > = 0, \text{for } i \neq j, \\ &< \Psi_i, \varphi_j > = 0, \text{for } i \neq j, \end{aligned} \tag{3.30}$$

Several examples of Daubechies' wavelets and scaling functions are shown in Figure 3-10. The order of the wavelet filter for orthogonal wavelets is always an even number. The order of each wavelet filter is shown by the indices used on the corresponding $\varphi(t)$ and $\psi(t)$ functions.

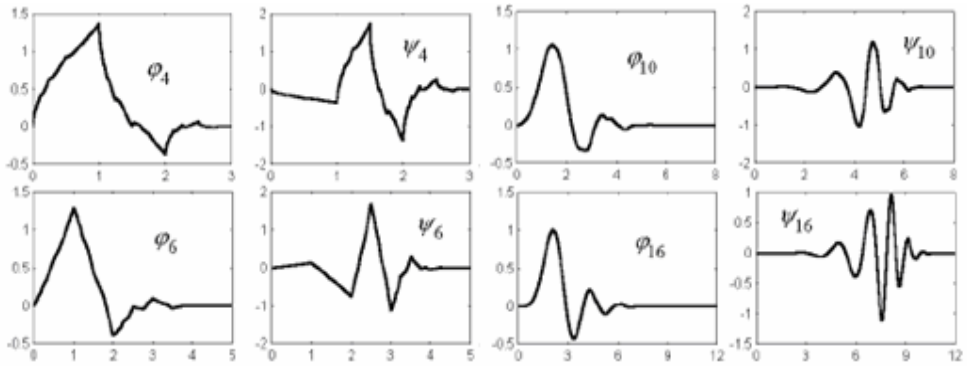


Figure 3-10 Examples of Daubechies scaling and wavelet functions

Daubechies wavelets have good compression property for wavelet coefficients but not for approximation coefficients. In other words, with reference to (3.21), the wavelet coefficients γ's have the best compression property under the circumstances while λ's do not [3]. In order to resolve this issue, Coifman suggested to have as many zero moments for scaling function as for wavelets [6]. This modification resulted in what is referred to as *Coiflets*.

3.7.3 Coifman Wavelets (Coiflets)

The Coifman wavelets were proposed by R. Coifman in 1989 [6]. One of the relevant characteristics of Coifman wavelets is that the scaling function $\varphi(t)$ is symmetric and vanishes fast as t goes to infinity. The following relations exist:

$$\begin{aligned} &\int \varphi(t)dt = 1, \\ &\int t^a \varphi(t)dt = 0,\ a = 1,....,N\text{-}1; \\ &\int t^a \psi(t)dt = 0,\ a = 1,....,N\text{-}1; \end{aligned} \tag{3.31}$$

where N is called the order of the Coifman wavelets. Figure 3-11 shows several of the examples in this case. The order of each wavelet filter is shown by the indices used on the corresponding $\varphi(t)$ and $\psi(t)$ functions.

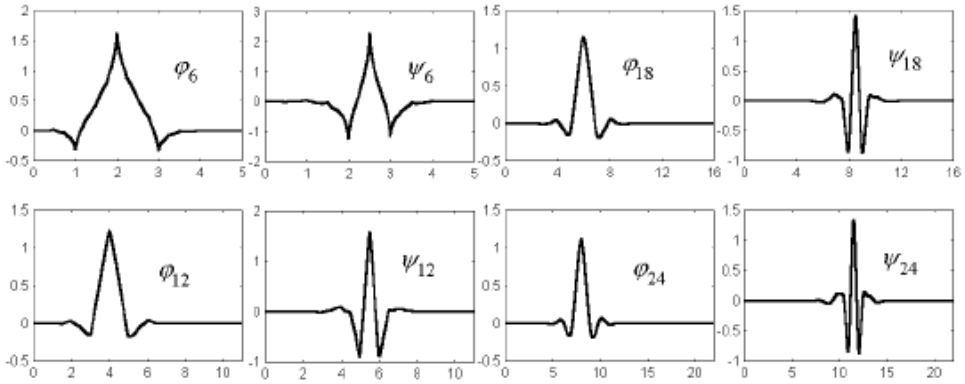


Figure 3-11 Examples of Coiflets scaling and wavelet functions

3.7.4 Burt Adelson

Burt Adelson wavelet is one of the well-known biorthogonal wavelets in wavelet literature. Because of its biorthogonal feature, it has two scaling and two wavelet functions where φ $\tilde{\varphi}$ and $\psi, \tilde{\psi}$ respectively.

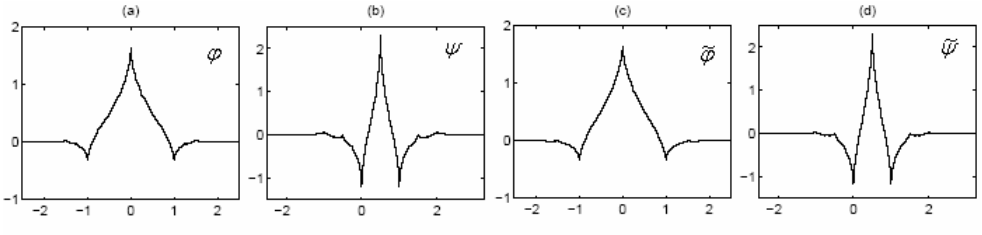


Figure 3-12 Example of two band Burt Adelson biorthogonal wavelet

Burt Adelson wavelets are generated by a one "real" parameter familiy of symetric filters with small support but enough regularity and vanishing moments.

3.7.5 Spline Wavelets

In Spline cases wavelet $\psi(t)$ and scaling $\varphi(t)$ functions are polynomial splines of degree n [9]. Spline wavelets can be classified to different types regarding to their orthogonality properties: othogonal, shift-orthogonal and biorthogonal are well known types.

Orthogonal spline wavelets:

The wavelets in this category were constructed independently by Battle and Lemarie [13] [14]. Battle-Lemaire wavelet is an example of orthogonal spline case. Wavelet and scaling functions of orthogonal spline wavelet represented in Figure 3-10 are the same due to orthogonality feature.

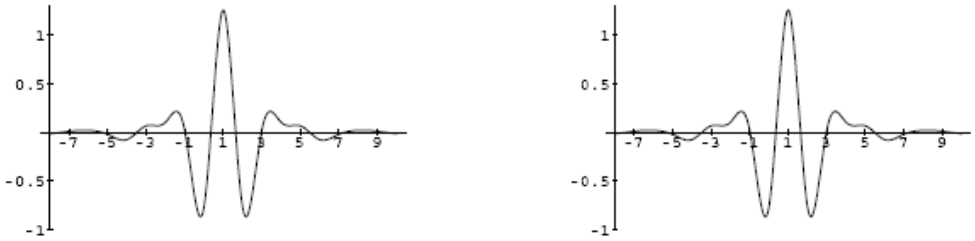


Figure 3-13 Wavelet (left) and scaling (right) functions of orthogonal spline wavelet.

Shift-orthogonal spline wavelets :

This category explores a last possibility which is to retain the intra-scale orthogonality requirement alone. Their main advantage is that it is possible to reduce the decay of the wavelet while essentially retaining the orthogonality and approximation properties of the Battle-Lemarié spline wavelets. These features can potentially be of interest for subband coding. Note that orthogonality property is required for the quantization error (see Chapter 3.6.1). Wavelet and scaling functions of shift-orthogonal spline wavelet are represented in Figure 3-11.

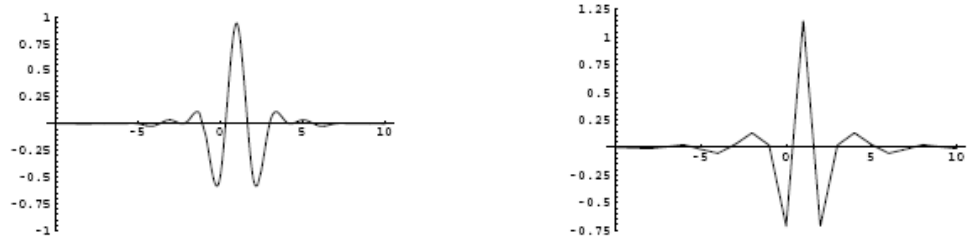


Figure 3-14 Wavelet (left) and scaling (right) functions of shift-orthogonal spline with (n=3, $\tilde{n}$ = 1) where n represents the degree of synthesis and analysis spline.

Shift-orthogonal splines wavelets are shorter synthesis filters. According to Lemarié this is advantageous for reducing reconstruction artifacts e.g., spreading of coding errors, ringing around sharp transitions [14].

Biorthogonal spline wavelets :

Biorthogonal wavelet basis shown in Figure 3-15 were introduced by Cohen-Daubechies-Feauveau (CDF) in order to obtain wavelet pairs that are symmetric, regular and compactly supported [8]. Biorthogonal wavelets build with splines are especially attractive because of their short support and regularity. These wavelets turn out to be quite popular for coding applications [9]. In particular, the symmetry and short support (compactly supported with in a short interval) properties are very valuable for reducing truncation artifacts in the reconstructed images.

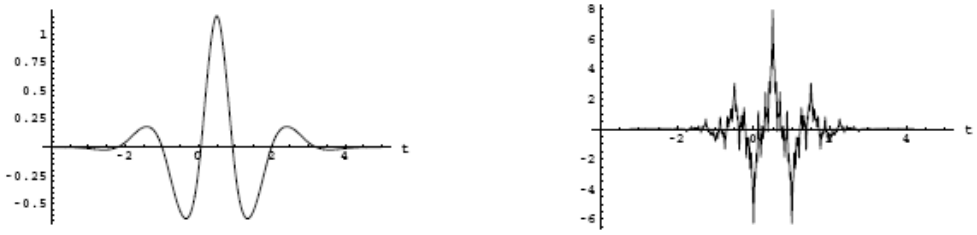


Figure 3-15 Wavelet (left) and scaling (right) functions of biorthogonal spline wavelet with ($L = 4,\ \widetilde{L} = 6$). L and $\widetilde{L}$ represent order of synthesis and analysis spline respectively.

3.8 Results

There exist many different wavelet bases with different characteristics. Success of a given wavelet basis in a particular application does not necessarily mean that this set is efficient in other applications. In the same manner success of a given wavelet basis can vary on different image types. Therefore, the freedom for choosing a particular wavelet should carefully be explored.

In previous chapter classification of wavelet models was introduced. Number of order and orthogonality feature of different wavelet bases may affect the quality of reconstructed image. In order to see this behaviour, comparison of the input image with the wavelet transformed and reconstructed final image have to be done. All wavelet models implemented in this work can be seen in Appendix B.

The Peak Signal to Noise Ratio (PSNR) is most commonly used as a measure of quality of reconstruction in image compression and image denoising works [16]. It comes from mean square error (MSE). MSE of two images are defined as

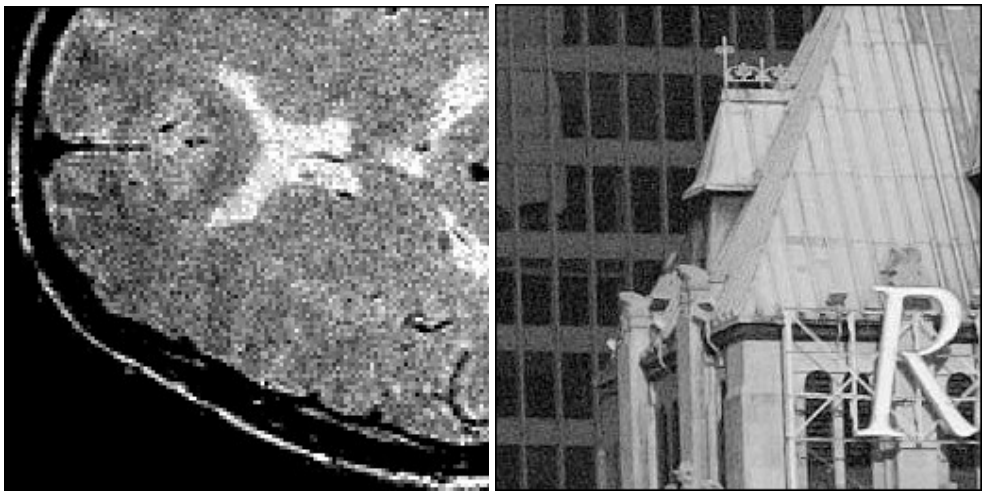
$$\text{MSE} = {}^{1}\!/_{mn} \left(\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (I(i,j) - R(i,j))^2 \right) \tag{3.11}$$

where I and R can be interpreted as input and reconstructed images respectively. m and n defines number of pixel in vertical and horizontal dimension of images I and R. Then the PSNR is defined as

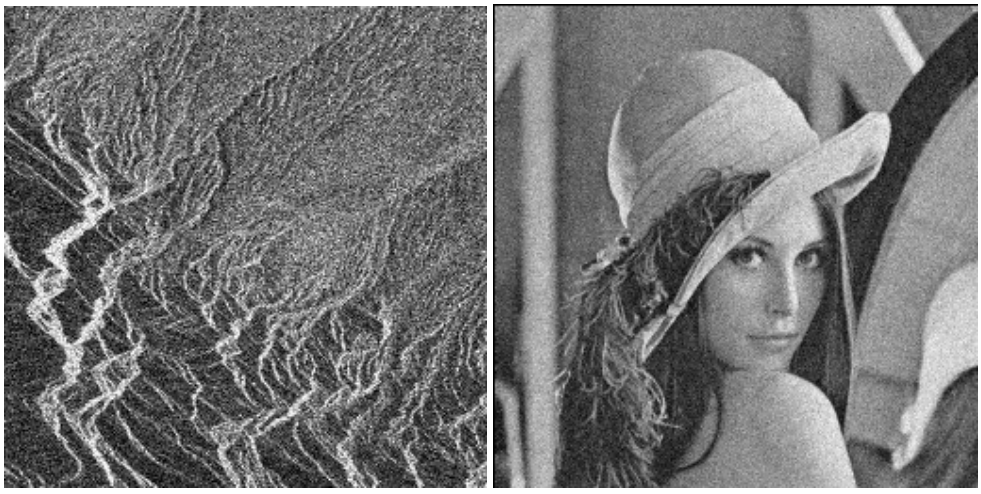
$$\text{PSNR} = 10.\log_{10}\left(MAX_I^2 / MSE\right) = 20.\log_{10}\left({}^{MAX_I}\!/_{\sqrt{MSE}}\right) \tag{3.12}$$

where MAX_I is the maximum pixel value of the image I. When the pixels are represented using 8 bits per sample "*grey scale"*, MAX_I takes value "255". More generally, MAX_I is 2^B-1 where B is a color dept "*bit"* of an image.

Comparing input image with reconstructed image (without denoising) is a good measure in order to evaluate success of wavelet reconstruction. PSNR of these two image can give significant information about quality of reconstruction. Higher PSNR value means less difference between input and reconstructed image and vice versa. Different 8-bit noisy images can be seen in Figure 3-16. First image, MRI, is an example of magnetic resonance imagery. Second image, Roof, is an ordinary image. Third image SAR is an example of Synthetic Aperture Radar imagery. Last one, Lena, is one of the well known and mostly used portraits.



a. MRI b. Roof



c. SAR image d. Lena

Figure 3-16 Several 8-bpp noisy images

PSNR values of all wavelet models on various images shown in Figure 3-16 were calculated. Results are shown in Table 3-1. If final reconstructed image is totally identical or has very minor difference with respect to input image than PSNR value goes to infinity.

Table 3-1 PSNR values of various wavelet models on different images

	MRI	Roof	SAR	Lena
Wavelet model	PSNR values			
Haar	81.25	Infinity	Infinity	Infinity
Daubechies 4	50.90	50.21	50.26	50.02
Daubechies 6	49.21	48.15	48.14	48.15
Daubechies 8	49.21	48.15	48.13	48.15
Daubechies 10	49.21	48.15	48.13	48.15
Daubechies 12	53.45	51.03	50.56	51.13
Daubechies 20	Infinity	Infinity	Infinity	Infinity
Coiflet 2	78.10	90.01	93.29	94.53
Coiflet 4	Infinity	Infinity	Infinity	Infinity
Coiflet 6	Infinity	Infinity	Infinity	Infinity
Pseudo Coiflet 4	86.75	86.75	Infinity	Infinity
Spline 2-2	84.25	Infinity	Infinity	Infinity
Spline 2-4	68.47	68.43	71.31	70.64
Spline 3-3	60.60	60.26	59.31	60.87
Spline 3-7	25.39	23.04	28.24	22.82
Battle Lemaire	48.31	47.41	50.58	47.13
Burt Adelson	79.13	89.09	86.29	93.28
CDF 2-4	52.91	52.42	53.03	52.47

3.9 Conclusion

Table 3-1 shows that various wavelet models have different results on final reconstructed image. Lets deal with Haar wavelet known as the most simple and symmetrical one in orthogonal case. Despite its simple structure it has very good results and computationally very fast indeed. Due to simplicity and existence of fast computational algorithm, Haar transform can be a good choice for image processing for specific situations especially when the performance is needed. But due to its simple structure how good this wavelet can be combined with denoising scheme will be investigated.

Next ones are Daubechies known as orthogonal wavelets which have more vanishing moments compared to Haar. It is seen that Daubechie with order 20 gives one of the best result in the test. Although other ones with lower orders are very close to each others, Daubechies with higher orders tend to give better results.

There is a same situation for Coiflet. Higher order Coiflets have better results. In general Coiflets are superior than Daubeshies and Splines. This is probably from its structure which has more symmetrical properties and more vanishing points compared to Daubechies and Splines. In overall Coiflets seem to be best for all images used in this test.

For the case of Spline, three types were compared in this test. Battle Lemaire filter known as orthogonal spline wavelet, CDF known as biorthogonal spline wavelet and shift-orthogonal spline wavelets “spline 2-2, spline 2-4, spline 3-3 and spline 3-7”. Between all of them spline 2-2 have the best results. Shift orthogonal splines have different behaviour with respect to other wavelet familes such as Daubechies or Coiflets. Shift orthogonal splines with lower orders tend to give better results. It is clearly seen that order of wavelet model has an impact on final image.

In this test there are two biorthogonal wavelet model used: CDF and Burt Adelson. Remember that biorthogonal wavelets have more symmetry than orthogonal ones. According to many wavelet works symmetry of wavelet is one of the most important feature for reducing truncation artifacts in the reconstructed images. Although Burt Adelson performed very well in this test, CDF had average results. Note that there was no denoising scheme on wavelet domain. In real world how good the performance of this wavelets will also be studied in next chapters.

Since the purpose of this work is comparing image denoising methods on the bases of wavelets, the next chapter is going to deal with denoising operations. Up to now only wavelet models have been compared on different images without any shrinkage or denoising. This could give some useful information about how wavelets act on different images. For example according to Table 3-1 Daubechies seem to have better result on MRI. Also Coiflets and Burt Adelson are clearly better for Lena. CDF and Battle Lemaire are more usable for astronomical image. For all images Daubechies and Coiflets with higher order are giving the best results.

CHAPTER 4

DENOISING OPERATIONS IN WAVELET DOMAIN

In previous chapter wavelet theory was studied and wavelet coefficients (WC) of noisy image are found using wavelet models was also introduced. To summarize, splitting the signal into equally sized coarse and detail values lies on the bases of WT. Coarse and detail representation of signal is shown in Expression 3.21. Mallat's decomposition tree in Figure 3-6 shows that in each iteration level of WT, low pass filter G_0, *scaling function*, produce coarse approximations, and high pass filter H_0, *wavelet function*, produces detail approximations. For the next level using output of low pass filter, coarse and detail approximations are produced again according to Expression 3.24. In each iteration level so called *scale*, size of WC is halved so it is seen that number of wavelet scale depends on the signal length. Suppose $X_1 \ldots\ldots X_n$ are noisy data where $n = 2^j$ for some integer $J > 0$. If this condition does not meet then n will be adapted to this condition which is mentioned in Section 2.4. Remember that, WC in different scales show image characteristics with different intensities. Because of the noise in the image, WC inevitably include information about noise.

4.1 Introduction

Decomposition of a function f into wavelet components can be represented as :

$$\gamma_{j,k} = \int_0^1 f(u)\psi_{j,k}(u)du \tag{4.1}$$

where $\gamma_{j,k}$ known as true wave coefficients is also called wavelet parameter values. In nonparametric regression (wavelet domain), the parameter values are unknown, so they must be estimated from the data. Empirical Wavelet Coefficients (EWC) are defined corresponding to the true coefficients $\gamma_{j,k}$ as follows:

$$w_{j,k}^{(n)} = \int_0^1 \widetilde{f}(u)\psi_{j,k}(u)du \tag{4.2}$$

where $\widetilde{f}()$ is the function estimated from data values $X_1 \ldots\ldots X_n$ which are distributed as $X_i \sim N(f(i/n), \sigma^2)$. It is relatively straightforward to derive the approximate distribution of $w_{j,k}^{(n)}$ from (4.2). In particular, $w_{j,k}^{(n)}$ is normal with mean

$$\begin{aligned} \mathrm{E}[w_{j,k}^{(n)}] &= \mathrm{E}\left[\int_0^1 \widetilde{f}(u)\psi_{j,k}(u)du\right] \qquad (4.3) \\ &= \sum_{i=1}^{n} E[X_i] \int_{(i-1)/n}^{i/n} \psi_{j,k}(u)du \\ &= \sum_{i=n}^{n} f(i/n)\psi_{j,k}(i/n) + O(\frac{1}{n^2}) \\ &= \int_0^1 f(u)\psi_{j,k}(u)du + O(\frac{1}{n}) + O(\frac{1}{n^2}) \\ &= \gamma_{j,k} + O(\frac{1}{n}) \qquad (4.4) \end{aligned}$$

The EWC $w_{j,k}^{(n)}$ is thus an estimator of the true coefficients $\gamma_{j,k}$ and it can be written in the form of (4.5) .

$$\sqrt{n}\left(w_{j,k}^{n} - \gamma_{j,k}\right) \sim N\left(0, \sigma^2\right) \tag{4.5}$$

where σ^2 is variance and N is the sample size.

As pointed out by Donoho and Johnstone, each set of EWC consist of a certain amount of noise, but only relatively few consist of significant signal [19]. The noise in the original sequence $X_1 \ldots X_n$ is spread out uniformly among all EWC [19]. The question is "Which of the coefficients contain significant signal, and which are mostly noise ? ".

4.2 Wavelet thresholding

Since the largest true coefficients $\gamma_{j,k}$ are the ones that should be included in a selective reconstruction, in estimating an unknown function it is natural to include only coefficients which have absolute value larger than some specified threshold value. Such a threshold value is found to distinguish between EWC that belong in the reconstruction (corresponding, one would hope, to true coefficients which contribute significant signal) and those that do not belong (corresponding to negligibility small true coefficients). Based on (4.5), the threshold estimator of the true coefficients $\gamma_{j,k}$ can be written as:

$$\hat{\gamma}_{j,k} = \frac{\sigma}{\sqrt{n}} \delta_{Tc}\left(\frac{\sqrt{n} w_{j,k}^{(n)}}{\sigma}\right) \tag{4.6}$$

where the function δ_{Tc} is called thresholding function, n is sample size and σ is noise level or in other words Standard Deviation (STD) of coefficients for specific level [16].

Here σ is estimated for wavelet coefficients. Donoho and Johnstone propose an estimate of the noise level that is based only on the EWC at the highest level. The reason for considering only highest level of coefficients is that these tend to consist mostly of noise [20].

$$\hat{\sigma} = \frac{median(|w_{j-1,k}^{(n)} - median(w_{j-1,k}^{(n)})|)}{0.6745} \tag{4.7}$$

where $J = \log_2(n)$. This process is called Median of Absolute Deviation (MAD).

Finding the threshold value T_c will be treated in the next sections. In this chapter, it is assumed that T_c is known already. The question is how the threshold value is applied. There are different approaches for thresholding or so called shrinking on the WC. Hard and soft thresholding schemes which were proposed and studied by Donoho and Johnstone are well known ones [19][20].

4.2.1 Hard thresholding

The procedure in which small EWC $w_{j,k}^{(n)}$ are removed while others are left untouched is called hard thresholding. Hard thresholding sets any coefficient less than or equal to the threshold T_c to zero.

$$\delta_{Tc}^{H} = \begin{cases} x, & if\ |x| > T_C \\ 0, & otherwise \end{cases} \tag{4.8}$$

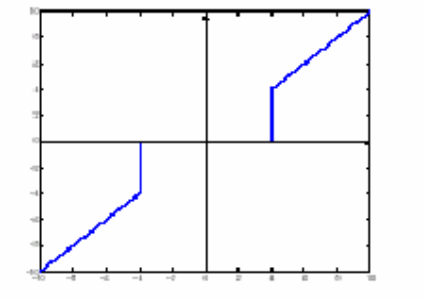


Figure 4-1 Transfer function of Hard thresholding

Hard threshold is a "keep or kill" procedure and is more intuitively appealing. This method generates spurious blips, better known as artifacts, in the images as a result of unsuccessful attempts of removing moderately large noise coefficients. Hard thresholding does not even work with some algorithms such as SureShrink (see Section 4.5.2.1). To overcome the demerits of hard thresholding, wavelet transform using soft thresholding was introduced.

4.2.2 Soft thresholding

In this scheme, EWC $w_{j,k}^{(n)}$ above the threshold T_c are shrunk by the absolute value of the threshold itself. Soft thresholding sets any coefficient less than or equal to the threshold to zero then threshold is subtracted from any coefficient that is greater than the threshold.

$$\delta_{Tc}^{S} = \begin{cases} x - T_C, & if\ x > T_C \\ 0, & if\ |x| \le T_c \\ x + T_C, & if\ x < -T_C \end{cases} \tag{4.9}$$

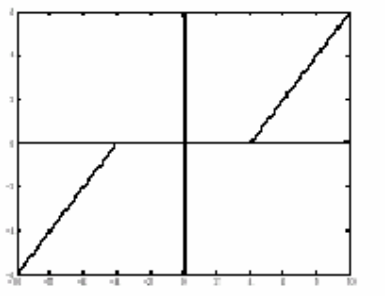


Figure 4-2 Transfer function of soft thresholding

While at first sight hard thresholding may seem to be natural, the continuity of soft thresholding has some advantages. Sometimes, pure noise coefficients may pass

the hard threshold and appear as annoying *blips* in the output. Soft thesholding shrinks these false structures.

4.3 Estimating of the Threshold

In many applications estimating of the threshold value is done for each wavelet scale individually. WC in each scale under this estimated threshold values are accepted as noise characteristic of image.

Threshold estimators or so called filtering operators in the wavelet domain can be subdivided into linear and nonlinear methods as shown in Figure 4-3.

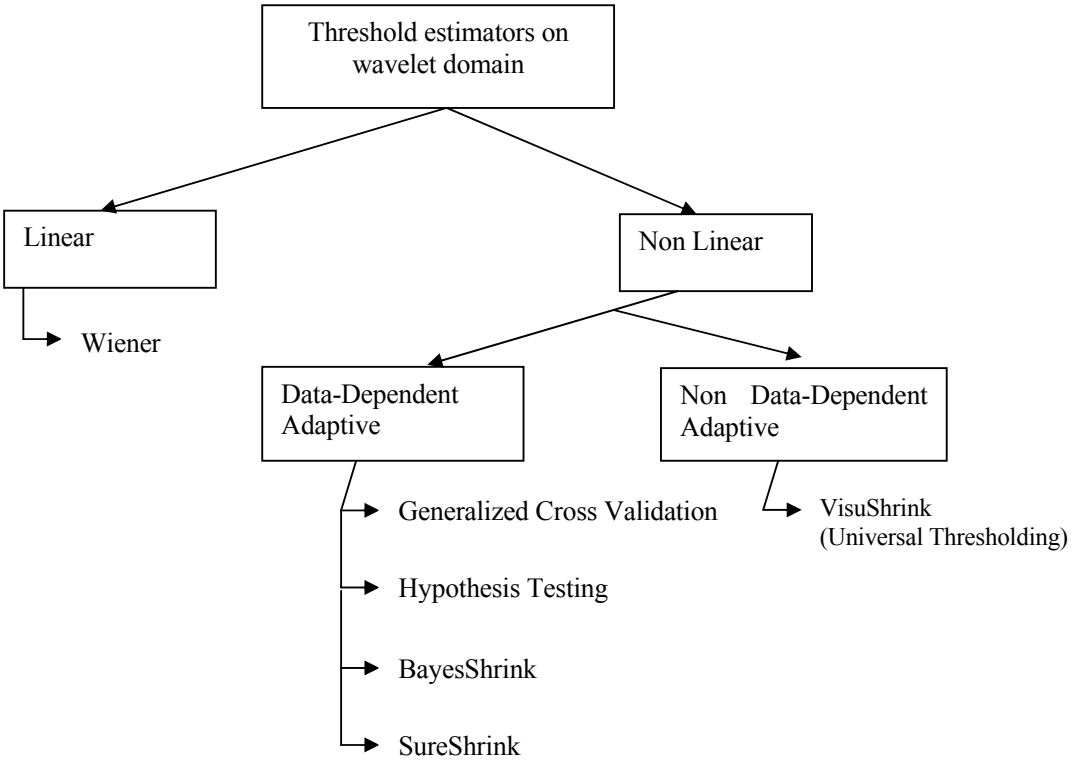


Figure 4-3 Schematic diagram of filtering operations in the wavelet domain

In this section, performances of distinct threshold estimators will be analyzed starting with fixed threshold value for all wavelet scales called universal thresholding which is the most primitive one and go on with the most advanced data adaptive statistical methods. Between these methods also the results of Wiener filtering will be tested. Wiener filtering is a kind of linear denoising algorithm which is very successful if the noise distribution of original noisy image is similar to Gaussian distribution with known standard deviation [16].

4.4 Linear Filters (Wiener Filter)

Linear filters such as Wiener filter in the wavelet domain can give good results if the signal corruption can be modeled as a Gaussian process [17]. But designing a filter based on this assumption frequently results in a filtered image that is more visually displeasing than the original noisy signal. In a wavelet-domain spatially adaptive finite impulse response (FIR) Wiener filtering for image denoising is proposed [18]. It means that filtering is performed only within each wavelet scale, in other words threshold are calculated and applied for each scale separately.

Algorithm of spatially adaptive FIR Wiener filter is giving :

$$T_{c_j} = \quad T_{m_j} \times \sigma_{S_j} \times \sigma_o \tag{4.10}$$

where T_{c_j} , T_{m_j} , σ_{S_j} and σ_o are threshold value, the threshold multiplier, the STD of wavelet coefficients of *simulated image* for each wavelet level j and the STD of the noise in noisy data respectively.

Because of the nature of linear filters, there is a need for a model to represent signal corruption of noisy image. This model with exactly the same size with original noisy image is called *simulated image* and created by adding noise to blank image using Gaussian process with STD “1”. Note that this model is created

independently and assumed that it has a similar noise characteristic with the original noisy image. Second step is to calculate WC of this simulated model. Remember that Wiener filter is executed in wavelet domain in this work.

The STD of the WC of simulated data is found by the MAD process described in Equation 4.7 however this value is calculated for each wavelet level here. STD of noise in the noisy data is estimated also by the same way.

$$\sigma_o = \text{STD}\ (|x_i - \tilde{x}|) \tag{4.11}$$

where x is noisy data, $\tilde{x}$ is the median of the noisy data and $|x|$ is the absolute value of x. Note that σ_o is calculated once and used for filtering operations in all scales.

STD of WC of simulated image "σ_s" is not a single unchangeable value. It is calculated for every wavelet scale of simulated noise model.

T_m is the standard deviation multiplier or in other words thresholding coefficient. It can be different in each scale. Value of T_m can be zero or any other positive number. When it is zero threshold value will become zero so there will be no denoising. Small values mean removing smaller noise.

4.5 Non-linear Filters

The most investigated domain in WT denoising is the non-linear coefficient thresholding based methods. The procedure exploits sparsity property of the wavelet transform and the fact that the WT maps white noise in the signal domain to white noise in the transform domain [19]. Thus, while signal energy becomes more concentrated onto fewer coefficients in the transform domain, noise energy

does not. It is this important principle that enables the separation of signal from noise [19].

4.5.1 Non Data-Dependent Adaptive Threshold (VisuShrink)

Non data ataptive thresholding is done with fixed threshold value called Universal Threshold (UT) seen in Equation 4.12.

$$T_c = \sigma\sqrt{2\ln N} \tag{4.12}$$

where T_c is threshold value, N is the data length, σ is the noise variance of data estimated according to Equation 4.11.

Universal thresholding is non-data dependent because it is not inspecting each data statistically. However it is certainly an adaptive threshold method due to parameters such as *N* and σ in its expression. Although universal thresholding is not the best method to determine a threshold, it can be useful to obtain a starting value when nothing is known about the signal condition. One can say that the universal threshold may give a better estimate for the soft threshold if the number of samples is larger since the threshold is optimal in the asymptotic sense.

VisuShrink proposed by Donoho and Johnstone is another name of UT applied on wavelet coefficients [19]. This threshold is given by :

$$T_c = \sigma\sqrt{2\log M} \tag{4.13}$$

where σ is the noise variance estimated according to (4.11) and M (NxN) is the number of pixels in the image . It is proved that the maximum of any M values, $N(0,\sigma^2)$ will be smaller than the universal threshold with high probability, with the probability approaching 1 as M increases. Thus, with high probability, a pure

noise signal is estimated as being identically zero. However, for de-noising images, it is found out that VisuShrink yields an overly smoothed estimate [19]. This is because UT is derived under the constraint that with high probability, the estimate should be at least as smooth as the signal. So the UT tends to be high for large values of M, killing many signal coefficients along with the noise. Thus, the threshold does not adapt well to discontinuities in the signal.

4.5.2 Data Dependent Adaptive Thresholding

A data-dependent analysis assumes that almost all the information available to a researcher is contained within the observed data. Data-dependent methods for the analysis of WC are shown to provide significant advantages over conventional techniques mentioned in previous sections. There are plenty of works available in this area. Well known methods are SURE shrinking, Bayesian shrinking Generalized Cross validation, and Hypothesis testing.

4.5.2.1 SURE Shrink

Donoho & Johnstone introduce a scheme that uses the coefficients at each wavelet level j to choose a threshold T_{c_j} with which to shrink the coefficients at that level which is known as the SureShrink wavelet thresholding technique [19, 20].

The basic idea behind this scheme is to find an estimator $\hat{f}$ for f that will have small L^2 risk :

$$R(\hat{f}, f) = E\left[\frac{1}{n}\sum_{i=1}^{n}\left(\hat{f}(i/n) - f(i/n)\right)^2\right] \tag{4.14}$$

This quantity can be expressed in terms of the WC,

$$R(\hat{f}, f) \propto E\left[\sum_j \sum_k \left(\hat{\gamma}_{j,k} - \gamma_{j,k}\right)^2\right] \tag{4.15}$$

where $\gamma_{j,k}$ are true wavelet coefficients described in Section 4.1.

The significance of this is that it is possible to transform the original data into its WC, and then attempt to minimize risk in the wavelet domain; doing so will automatically minimize risk in the original domain.

In practical situation the risk $R(\hat{f}, f)$ must be estimated from the data. This method employs an unbiased estimate of risk that is due to Stein called Stein Unbiased Risk Estimator (SURE) [27].

Minimization of estimated risk is done by choosing a threshold value for each wavelet scale. This method is illustrated by considering the following equivalent problem. Suppose $(X_1 \ldots X_d)$ are independent observations with $X_k \sim N(\mu_k, 1)$. The problem is to estimate the mean vector $\mu = (\mu_1 \ldots \mu_d)'$ with minimum risk. This represents estimation of true WC at any level j, with $X_k = \sqrt{n} w_{j,k}^{(n)}$ and $d = 2^j$

Since it is thought that most of the coefficients are zero, the estimator will be the soft thresholding function (see Section 4.4.2). Denoting the vector of observation X and letting $\hat{\mu}$ represent the resulting estimator of μ, the result of Stein states that the l^2 loss can be estimated unbiasedly for an estimator of μ that can be written $\hat{\mu}(X) = X + g(X)$ where the function g : $IR^d \to IR^d$ is weakly differentiable :

$$E_\mu \left\| \hat{\mu}(X) - \mu \right\|^2 = d + E_\mu \left\{ \left\| g(X) \right\|^2 + 2\nabla . g(X) \right\} \tag{4-16}$$

where $\nabla.g \equiv \sum_{k=1}^{d} \frac{\partial}{\partial X_k} g_k(X)$, defining $g = (g_1 ... g_d)$. Using the soft thresholding function δ gives that

$$g_k(X) = \delta_t(X_k) - X_k = \begin{cases} -t, & X_k \geq t, \\ -X_k, & -t \leq X_k < t, \\ t, & X_k < -t \end{cases} \tag{4.17}$$

so $\|g_k(X)\|^2 = \sum_{k=1}^{d} \min^2(|X_k|, T_c)$. Note also that $\nabla.g = -\sum_{k=1}^{d} 1_{[-T_c, T_c]}(X_k)$, so that Stein's estimate of risk applied to this situation can be written for any set of observed data $x = (x_1 ... x_d)'$:

$$\begin{aligned} \text{SURE}(T_c, x) &= d - 2.\#\{k : |x_k| \leq T_c\} + \sum_{k=1}^{d} \min^2(|x_k|, T_c) \\ &= -d + 2.\#\{k : |x_k| > T_c\} + \sum_{k=1}^{d} \min^2(|x_k|, T_c) \end{aligned} \tag{4.18}$$

where #S for a set S denotes the cardinality of the set. Here, $E_\mu \left\| \hat{\mu}^{(T_c)}(X) - \mu \right\|^2 = E_\mu SURE(T_c; X)$.

The threshold level is set so as to minimize the estimate of risk for the given data $(x_1 ... x_d)$:

$$T_c = \arg\min_{t \geq 0} SURE(t; x) \tag{4.19}$$

Such a method can reasonably be expected to do well in terms of minimizing risk, since for large sample sizes the Law of Large Numbers will guarantee that the SURE criterion is close to the true risk [23].

4.5.2.2 Generalized Cross Validation

Cross-validation is widely used as an automatic procedure to choose the smoothing parameter in many statistical settings. The generalized cross-validation (GCV) method is performed by systematically expelling a data point from the construction of an estimate, predicting what the removed value would be and, then, comparing the prediction with the value of the expelled point [25].

Based on the work of Nason, G.P., the minimizer of the GCV function for threshold selection has been used [26]. Base of GCV function depends on choosing a global threshold to minimize L^2 risk in recovering the unknown function f. Thus the objective function is given by

$$M(t) = E\left[\int_0^1 \left(\hat{f}_t(x) - f(x)\right)^2 dx\right] \tag{4.20}$$

where $\hat{f}_t$ is the wavelet estimator that result from applying a threshold *t* globally to all WC. In practice, this can never be computed (as *f* is unknown), so *M(t)* must be estimated in some way. Discretized version of Expression 4.15 can be written as :

$$m(t) = E\left[\sum_{i=1}^{n} \left(\hat{f}_t(i/n) - f(i/n)\right)^2\right] \tag{4.21}$$

This can be expressed in the wavelet domain as :

$$m(t) = E\left[n\sum_j \sum_k \left(\hat{\gamma}_{j,k} - \gamma_{j,k}\right)^2\right] \tag{4.22}$$

where $\hat{\gamma}_{j,k}$ represents the estimate of the true coefficients $\gamma_{j,k}$ described in Section 4.1 computed from the soft thresholding function with threshold t :

$$\hat{\gamma}^{t}_{j,k} = \frac{1}{n}\delta_t(\sqrt{n}w^{(n)}_{j,k}) \tag{4.23}$$

The first question of interest in this approach involves the behavior of the objective function $m(t)$: In particular, given knowledge of f (equivalently, given knowledge of the $\gamma_{j,k}$'s) the question is "*is it possible to minimize the function* $m(t)$ *?*" [23]. Standard calculus concepts would suggest taking the first derivative of $m(t)$, solving for the t_0 which gives $m'(t_0)$ = 0, then verifying that t_0 is indeed a minimum by checking the sign of the second derivative of m at t_0. Unfortunately, this cannot be used, as the soft thresholding function δ_t applied to data values is not everywhere differentiable with respect to *t*. Nason examines this question at length, noting that the derivative $m'(t)$ is a piecewise linear with discontinuities at each $|\sqrt{n}w^{(n)}_{j,k}|$ [26]. With high probability, however, the jumps at these points are small in the area where the minimum is likely to occur. By this and additional arguments about the behavior of $m(t)$ for *t=0,* Nason concludes that the function $m'(t)$ is "almost convex" and can thus be effectively minimized [26].

The usual method of cross validation cannot be applied directly to estimation with wavelets because efficient algorithms are not yet available for computing the discrete wavelet transform using an orthogonal wavelet with non-uniform designs. Nason suggests breaking the original data set $(X_1 \ldots X_d)$ into two subsets of equal size: one containing only the even indexed data, and the other, the odd indexed data. The odd data will be used to "predict" the even data, and vice versa.

Let $(X^o_1, X^o_2, ..., X^o_{n/2})$ represent the *re-ordered* odd data points and $(X^E_1, X^E_2, ..., X^E_{n/2})$ the similarly renumbered even data values. The usual wavelet

estimator based on the even-indexed points, denoted $\hat{f}^E$, consist of estimates of the function f at the points $\frac{2}{n},\frac{4}{n},...\frac{n-2}{n},1$; that using the odd data points $\hat{f}^O$ estimates f at $\frac{1}{n},\frac{3}{n},...\frac{n-1}{n}$. To compare these estimated points directly with original data values, interpolated versions of each data set are employed as :

$$\widetilde{X}_i^O = \begin{Bmatrix} \frac{1}{2}(X_{2i-1} + X_{2i+1}), & i = 1,...,\frac{n}{2}-1, \\ \frac{1}{2}(X_{n-1} + X_1), & i = \frac{n}{2} \end{Bmatrix} \tag{4.24}$$

for the odd data, and

$$\widetilde{X}_i^E = \begin{Bmatrix} \frac{1}{2}(X_n + X_2), & i = 1, \\ \frac{1}{2}(X_{2i-2} + X_{2i}), & i = 2,...,\frac{n}{2} \end{Bmatrix} \tag{4.25}$$

for the even data. Note that the indices on these interpolated versions of the data coincide with the indexing of the subsets of the data, and hence, of the estimators resulting from the subsets. Also, note that the interpolation scheme described above wraps the data around interval [0,1], which corresponds to periodic boundary handling. The cross validation approach will minimize the fallowing estimate of $m(t)$ [25] :

$$\hat{m}(t) = \sum_{i=1}^{n/2} \left\{ \left(\hat{f}_t^E\left(\frac{2i}{n}\right) - \widetilde{X}_i^O \right)^2 + \left(\hat{f}_t^O\left(\frac{2i-1}{n}\right) - \widetilde{X}_i^E \right)^2 \right\} \tag{4.26}$$

Defining for the moment $\{w_{j,k}^E\}$ and $\{w_{j,k}^O\}$ to be the wavelet coefficients for the even and odd-indexed data respectively, with $\{\widetilde{w}_{j,k}^E\}$ and $\{\widetilde{w}_{j,k}^O\}$ representing the

WC resulting from the respective interpolated sequences, the expression (4.21) can be rewritten as :

$$\hat{m}(t)=n\sum_{j}\sum_{k}\left\{\left(\frac{1}{\sqrt{n}}\delta_t\left(\sqrt{n}w_{j,k}^{E}\right)-\tilde{w}_{j,k}^{O}\right)^2+\left(\frac{1}{\sqrt{n}}\delta_t\left(\sqrt{n}w_{j,k}^{O}\right)-\tilde{w}_{j,k}^{E}\right)^2\right\} \qquad (4.27)$$

This function can then be minimized over *t*, giving the "best" threshold for half sample prediction :

$$T_{CV}^{n/2} = \arg\min_{t\geq 0}\hat{m}(t) \qquad (4.28)$$

The selected threshold (4.28) is not the best to apply to the full data set, however. Remember the dependence of the universal threshold of Donoho and Johnstone. In Expression 4.8 with sample size M : $T_{c_u} = \sigma\sqrt{2\log M}$.

Since $T_{T_u}^{n/2} = \sqrt{2\log(n/2)}$, the relationship between the two thresholds is

$$T_{C_U}^{n} = \left(1-\frac{\log 2}{\log n}\right)^{-\frac{1}{2}} T_{C_U}^{n} \qquad (4.29)$$

The final threshold to use with the full data set results from applying the Expression (4.24) to the cross-validation threshold for the half-samples [23].

Standard GCV algorithm taken from Nason was modified to compute a separate threshold for each resolution level in this work [26]. This gives more adaptive results but still there may be some problems in low level coefficient sets. Low level coefficient sets include most important part of image characteristics. Capturing and applying GCV low level coefficient sets may be resulted with loss of image characteristics.

4.5.2.3 Hypothesis Testing

The primary goal in the data dependent threshold selection is the division of WC into a group of "small" coefficients (those containing primarily noise) and one of "large" coefficients (those containing significant signal). A reasonable way for a statistician or a data analyst to go about this is to utilize statistical test of hypotheses, the large coefficients group consisting only of coefficients that "pass the test" of significance. This is the general approach taken by Odgen and Parzen [23, 24].

The general approach taken by Odgen and Parzen operates on a level-by-level basis, as does the SURE approach [23, 24]. At any particular level a single test is performed to determine if the set of coefficients at that level behave as white noise (small coefficents) or if there is a significant signal available. If it is determined that there is a signal present, the most likely candidate (the largest coefficients in absolute value) is removed from consideration, and the rest is repeated. Continuing recursively, at each level one will be left with two sets of coefficients : "large" coefficients thought to contain significant signal, and a set of "small" coefficients which is indistinguishable from pure white noise.

Let $X_1,.....,X_d$ represents the EWC $w_{j,k}^{(n)}$ at level $j = \log_2 d$ and suppose that these coefficients have means $\mu_1,.....,\mu_d$ respectively. Initially, interest is in testing the null hypothesis that all the means are zero vs. a general alternative that some of the μ_i's are non-zero. Specifically, let I_d represent a non-empty subset of the indices {1,…,*d*}. Then the hypothesis could be expressed as

$$\begin{aligned} &H_0 : \mu_1 = ... = \mu_d = 0 \\ &H_a : \mu_i \neq 0 \ \ for\, all\, i \in I_d ; \mu_i = 0 \ for\, all\, i \notin I_d \end{aligned} \tag{4.30}$$

where H_o is null hypothesis and H_a is alternative hypothesis.

A fundamental question that must be addressed in this approach is how to test the above set of hypothesis. p value which is the probability that a test statistic at least as significant as the one observed would be obtained assuming that the null hypothesis were true. The smaller the value of p , the stronger the evidence against the null hypothesis.

$$p = F(x \mid v) = \int_0^x \frac{t^{(v-2)/2} e^{-t/2}}{2^{v/2}\Gamma(v/2)} \tag{4.31}$$

Function in (4.31) is called Chi-square distribution where Γ() is the Gamma function [28]. The result, p is the probability that a single observation from a χ^2 distribution with ν degrees of freedom will fall in the interval [0,x]. The χ^2 density function with ν degrees-of-freedom is the same as the gamma density function with parameters ν/2 and 2.

Value of p is compared to an acceptable significance value α . If $p \leq \alpha$, that the observed effect is statistically significant, the null hypothesis is ruled out, and the alternative hypothesis is valid.

$$\alpha = 1 - p \tag{4.32}$$

Subtracting p from positive integer "1" gives the significance probability. As the probability gets smaller truth of hypothesis gets doubtful. If the probability is extremely low like "0.0001", it can be say that original assertion was wrong. If the probability is large "0.20" or greater than original assumption was reasonable.

This method operates by testing the maximum of each set of squared wavelet coefficients to see if it behaves as the nth order statistic of a set of independent

Chi-squares. If not, it is removed, and the max of the remaining subset is tested, continuing in this fashion until the max of the subset is judged not to be significant. If there is no significance at the end T_C is set to zero. If there is, T_C takes the value of this wavelet coefficent.

4.5.2.4 BAYES Shrink

Bayes rules are shrinkers and their application on wavelet coefficients leads to effective estimators of unknown signals [22]. The proposed Bayesian model relies on Bayesian Adaptive Multiresolution Shrinker (BAMS), a technique recently proposed by Vidakovic & Ruggeri [30]. It is based on Bayesian estimation using each EWC $w_{j,k}^{(n)}$ to estimate the corresponding true WC $\gamma_{j,k}$.

Bayesian methods for function estimation with wavelets are different then simple threshold selection, in the sense that new shrinkage functions result from the Bayesian approach, different from either the soft or hard thresholding functions discussed previously.

Let $X_1,.....,X_d$ represents the EWC $w_{j,k}^{(n)}$ at level $j = \log_2 d$. Each coefficient X is affected by normal errors, and thus the conditional distribution of X given θ and σ^2, $\mathrm{d}|\theta,\sigma^2|$ is $N(\theta,\sigma^2)$. In BAMS, the prior distributions on σ^2 and θ are chosen to be, respectively, an exponential one, $\Sigma(\mu)$, and a mixture of a point mass at zero and a double exponential distribution, $\partial\,\Sigma\,(0,\tau)$.

The Bayes rule for each X in the set $X_1,.....,X_d$ is :

$$\gamma_k = \frac{(1-\varepsilon)m(X_k)\delta(X_k)}{(1-\varepsilon)m(X_k) + \varepsilon\partial\Sigma(0,1/\sqrt{2\mu})} \tag{4.33}$$

where

$$m(X_k) = \frac{\tau e^{-|c_k|/\tau} - (1/\sqrt{2\mu})e^{-\sqrt{2\mu}|c_k|}}{2\tau^2 - 1/\mu} \tag{4.34}$$

It is seen that Bayesian rule is not a threshold estimator only. It is directly estimating γ_k without using soft or hard thresholding for a specific level. The rule is close to a thresholding rule because it heavily shrinks small-in-magnitude arguments with minor influence on the larger arguments [23].

4.6 Results

In this section various noise free images and artificially noise added versions are used for wavelet denosing filters mentioned in Section 4.4 and Section 4.5. Note that noise added to images is known as *Gaussian noise* or so called *white noise*.

Results of Wiener Filter :

Daubechie-20 wavelet model is used on noisy image shown in Figure 4-4 which is a noisy version of the original image with additive Gaussian noise. Denoising operation in wavelet domain is done only in the first three scale.

a. Original image

b. Noisy img. σ_g = 20, PSNR = 20.22

c. Hard, T_m (3,2,1) PSNR = 26.95

d. Soft T_m (1,0.5,0.25) PSNR = 26.95

Figure 4-4 Results of Wiener Filter

Wiener filtering in both cases hard and soft thresholding works well for the image used, if T_m is carefully selected for each scale. High T_m values result in loss of some detail especially in soft one whereas low T_m values result in less smoothing. Remember that success of Wiener filtering depends on how the signal corruption looks like the model of Gaussian process. If signal corruption can be modeled exactly, Wiener filter gives the best result in all wavelet denoising scheme without any doubt. But this is quite unusual.

Results of VisuShrink :

Daubechie-20 wavelet model is used on the noisy image shown in Figure 4-5 which is a noisy version of the original image with additive Gaussian noise. Denoising operation in wavelet domain is done only in the first three scale.

a. Original image

b. Noisy img. σ_g = 20, PSNR = 20.22

c. Visu, PSNR = 21.45

d. Visu, T_m (2,1,0.5) PSNR = 27.85

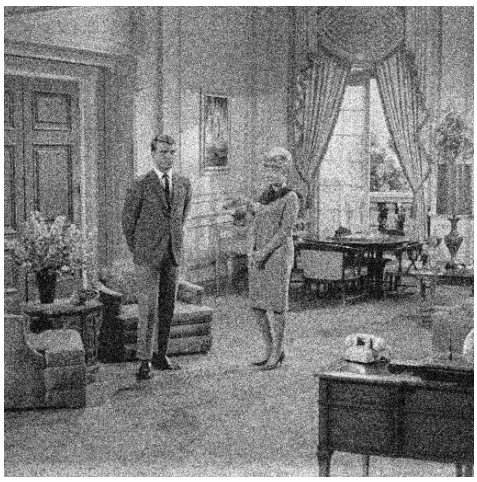
Figure 4-5 Results of VisuShrink

It is easly seen that hard thresholding is more convenient than soft one for VisuShrink. In both cases soft and hard small denosing coefficient multipliers were selected. Remember that VisuShrink uses the universal threshold. Although this threshold is in normal level for the first scale it is very high for the upper scales. Look at the last image with no multiplier. In that case universal threshold was applied in all scale without any limitation, so it yields overly smoothed image. Results show that VisuShrink is not a good way for wavelet denoising.

Results of SureShrink :

CDF 2-4 wavelet model is used on the noisy image shown in Figure 4-6 which is a noisy version of the original image with additive Gaussian noise. Denoising operation in wavelet domain is done only in the first three scale.

a. Original image



b. Noisy img. σ_g = 15, PSNR = 24.6

c. SURE , PSNR = 28.31

d. SURE + AM + DF, PSNR = 28.64

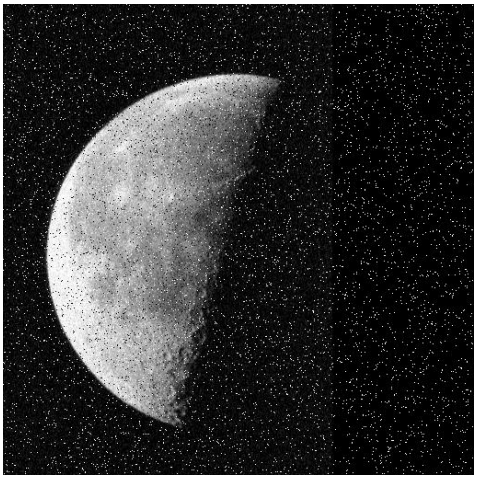
Figure 4-6 Results of Sure Shrink

It is seen that hard thresholding is not appropriate for SureShrink. Because significant denoising did not occur in hard case. On the other side soft thresholding gives very good results while preserving lots of image detail. For the last image (d) besides the soft SureShrink, DF over adaptive AM filtering with "σ_i = 1.6" was executed (see Chapter 5). It is seen image is slightly more superior due to small amount of positive contribution achieved using DF-AF filtering.

Results of Generalized Cross Validation :

CDF 2-4 wavelet model is used on the noisy image shown in Figure 4-7 which is a noisy version of the original image with additive Gaussian noise plus speckles or so called Salt&Pepper type noise. Denoising operation in wavelet domain is done only in the first two scale.

a. Original image



b. Noisy image σ_g =20,PSNR = 16.18

c. GCV (Hard 3 scale) + AM + DF
PSNR = 31.98

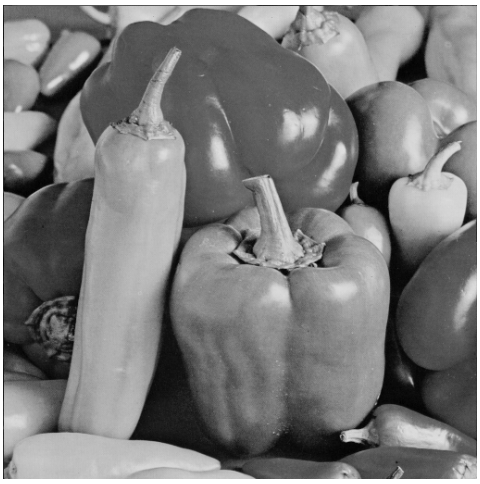
d. GCV (Soft, 2 scale) + AM + DF,
PSNR = 31.90

Figure 4-7 Results of GCV

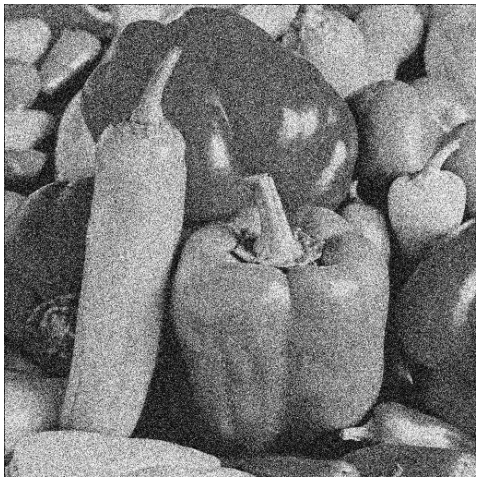
Both hard and soft thresholding methods are used here. Remember that original GCV was modified for level dependence. Modified GCV gives good result in both hard and soft thresholding case. Note that although GCV with soft thresholding was executed only for first two scale it gave the same good result as well as hard threshold did for first three scale. However several attempts for denoising higher resolution levels resulted with loss of image detail especially for soft thresholding. It is seen that GCV works perfect with lower resolution levels but it fails in higer resolution levels. This is not a serious problem. The important thing is how the wavelet denoising schemes act on lower levels where most of the noise are expected. Like in Sure thresholding, AM + DF filters are used here. It is seen image is slightly more superior due to small amount of positive contribution achieved using DF-AF filtering. For detailed information about these filters see Chapter 5.

Results of Hypothesis Testing :

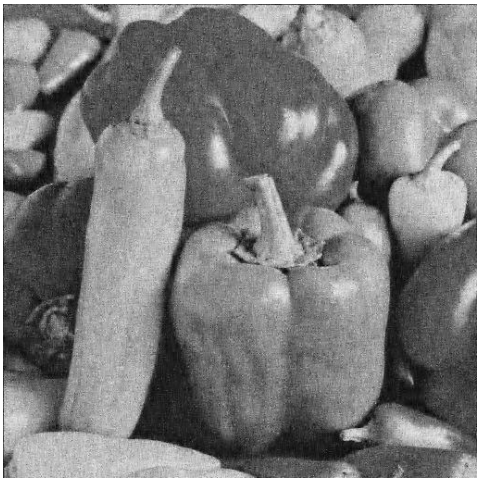
Spline 2-2 wavelet model is used on the noisy image shown in Figure 4-8 which is a noisy version of the original image with additive Gaussian noise. Denoising operation in wavelet domain is done only in the first three scale.



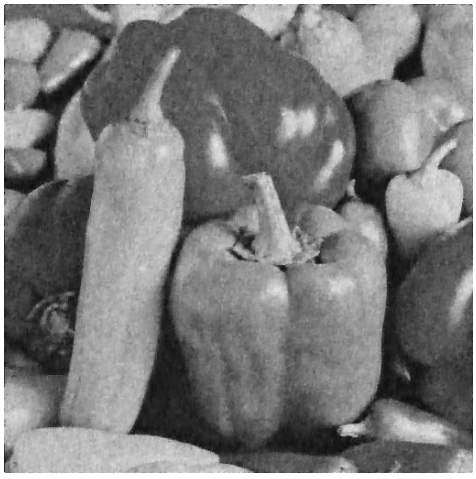
a. Original image



b. Noisy img. σ_g = 30, PSNR = 18.40



c. Hypo. , PSNR = 23.48, α = 0.5
Hypo., PSNR = 24.93, α = 0.05



d. Hypo., α = 0.05 , + AM + DF, PSNR = 26.50

Figure 4-8 Results of Hypothesis Test

Although thresholding with very minimum α values does work here for hard case, all tests were done with soft thresholding using two different α values "0.5" and "0.05" because of its better results with soft thresholding. For the last image (d) besides Hypothesis Test, double filtering (DF) over adaptive median (AM) filtering (σ_i = 1.6) was executed. It is seen image is slightly more superior due to small amount of positive contribution achieved using DF-AF filtering. For detailed information about these filters see Chapter 5.

The level of the hypothesis tests, significance level α has two values "0.5" and "0.05". It is seen that the choice of alpha controls the smoothness of the resulting wavelet estimator. In general, a relatively large alpha makes it easier to include coefficients, resulting in a more wiggly estimate; a smaller alpha will make it more difficult to include coefficients, yielding smoother estimates. Results show that smaller α equal to "0.05" gives better result then "0.5".

Results of Bayes Shrink :

CDF 2-4 wavelet model is used on the noisy image shown in Figure 4-9 which is a noisy version of the original image with additive Gaussian noise. Denoising operation in wavelet domain is done in all scales except for highest one.

a.Original image

b. Noisy img. σ_g = 20, PSNR = 21.64

c. BAYES, PSNR = 21.77

d. BAYES + AM + DF, PSNR = 22.40

Figure 4-9 Results of Bayes Shrink

Remember that Bayesian Shrink uses its own shrinkage method. In addition wavelet scale selection was disabled for this process. This means Bayesian Shrink works for all wavelet coefficients except for highest scale. Very high detailed image was used in this test in order to see response of this method which uses higher coefficient levels also. Look at the results seen in Figure 4-9. Denoising performance of Bayesian Shrink is reasonable but not very good. However it preserved almost all of the detail. It seems that Bayesian Shrink is perfect for denoising especially small-in-magnitude noise without smoothing. It sounds very good if preserving the image detail is very important. It is seen that image is slightly more superior due to a small amount of positive contribution achieved using DF and AF filtering. For detailed information about these filters see Chapter 5.

In this chapter various denoising operations have been studied on wavelet domain. In the next chapter some basic non-wavelet denoising methods will be introduced.

CHAPTER 5

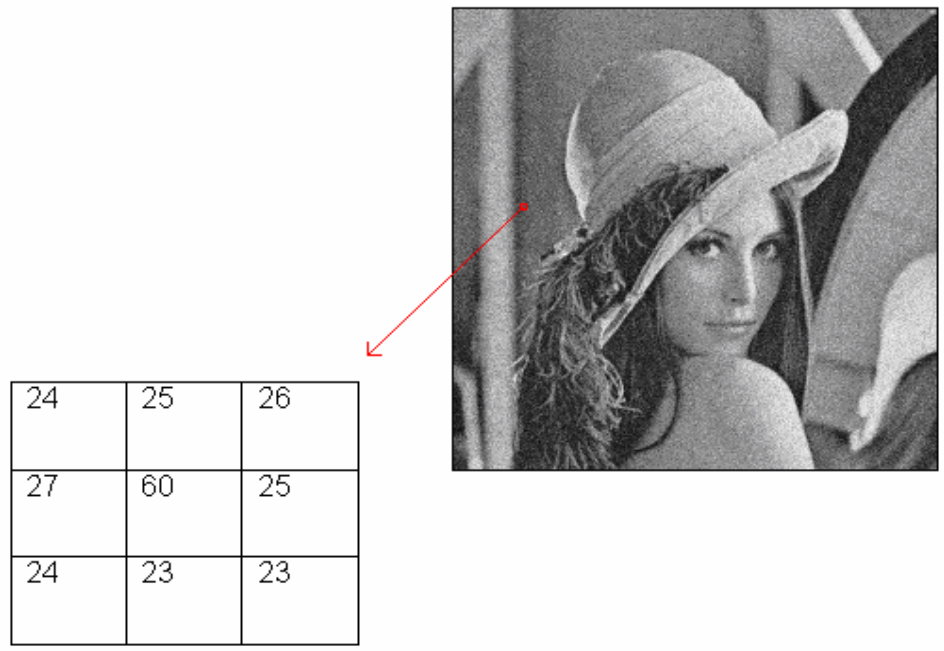
FURTHER PROCESES

The wavelet filters mentioned in previous chapter can be used for removing random and Gaussian-type noise, but it can leave some kind of speckle noise, *very hot pixels*. In wavelet transformation very hot pixels are assumed as detail values and other ones below low pass filter are assumed as coarse approximations (see Section 3.1). All wavelet based denoising filters discuessed in Chapter 4 are executed on coarse approximations so another methods are needed for denoising speckles. This can be provided in the form of non-wavelet filters. The idea behind these filters is that if an adequate sampling was chosen upon acquisition, no such outlying pixels with extreme values should be found. This idea can be useful for many image types. Note that all filtering operations in this chapter are directly related with real pixel values not on wavelet domain. In this chapter also a new tecnique called double filtering is introduced.

5.1 Median and Mean Filtering

Median filter is a kind of non linear filter to reduce noise in an image [32]. Among other applications, a median filter can be a useful tool for noise reduction. Why? There is a type of noise, known as *impulsional noise*, that is characterized by bright and/or dark high-frequency (small in relative scale) features appearing randomly over the image. Statistically, impulsional noise falls well outside the peak of the distribution of any given pixel neighborhood, so the median is well suited to learn where impulsional noise is not present, and hence to remove it by exclusion.

The idea is to examine a sample of the input and decide if it is representative of the signal. This is performed using a window which also be referred as "kernel" consisting of an odd number of samples. The size of window can be "3x3" pixel or more. In this work "3x3" are used default. The values in the window are sorted into numerical order; the median value, the sample in the center of the sorted list, is selected as the output. The oldest sample is discarded, a new sample acquired, and the calculation repeats.



24	25	26
27	60	25
24	23	23

Figure 5-1 Calculating the median value of a pixel neighborhood.

List of sorted pixel values of window is "{23,23,24,24,**25**,25,26,27,60}". Median value med_w is the center one and it is equal to "25". In this example in Figure 5-1, according to median filter operation center value of window x_i which is equal to "60" is replaced with med_w [32].

Logic of mean filter is same with median filter. Only difference is that x_i is replaced with mean value of window $mean_w$ instead of med_w.

5.2 Adaptive Median Filtering

Adaptive median filter is a data adaptive version of median filter. This filter will detect pixels that differ from their context by more than a given multiple of the neighborhood's standard deviation. If marked as outlying, the pixel value is replaced by the median value of the neighborhood. According to adaptive median filtering process :

$$\hat{x}_i = med_w \quad \text{if } \{ med_w > mean_w + \sigma_i \times \sigma_w \}$$
$$\text{if } \{ med_w < mean_w - \sigma_i \times \sigma_w \} \tag{5.1}$$

where med_w , $mean_w$ and σ_w is median value, mean value and STD of the kernel window respectively shown in Figure 5-1. σ_i is input standard deviation "user parameter" [29].

Finding the med_w is completely the same by the way of standard median filtering mentioned in Section 5.1 but when x_i is being replaced, neighborhood's STD value σ_w and a user parameter σ_i are taken account.

5.3 Lee Filter

The Lee filter is based on a linear speckle noise model and a minimum mean square error (MMSE) design approach [35]. It identifies regions with low and constant variance as areas for noise reduction.

In a region with no signal activity, the filter outputs the local mean. When signal activity is detected, the filter passes the original signal through unchanged. This is achieved by implementing a filter of the form specified by equation (5.2). Like

filters mentioned in Section 5.1 and Section 5.2 Lee filter also uses 3x3 local regions called kernel.

$$\hat{x}_i = W * center_w + (1 - W) * mean_w \tag{5.2}$$

where $center_w$ and $mean_w$ is the central pixel and mean value of 3x3 kernel respectively shown in Figure 5.1. W is the weighting function ranging between 0 for at regions to 1 for regions with high signal activity. The weighting function is calculated according to equation (5.3).

$$W = 1 - \frac{C_u^2}{C_I^e} \tag{5.3}$$

where $C_u = \frac{\sigma_u}{\bar{u}}$ and $C_I = \frac{\sigma_I}{\bar{I}}$ are the coefficient of variation of the noise u and the image I. σ represents the STD.

In the area where high variance are available, edges are assumed and little to no noise smoothing is done. In other words, the Lee filter smoothes away noise in this region, but leaves fine details unchanged. Therefore, its major drawback is that it leaves noise in the vicinity of edges and lines [38].

5.4 Kuan Filter

In the Kuan Filter, the multiplicative noise model is first transformed into a signal-dependent additive noise model [36]. Then the MMSE criterion is then applied to this model. The resulting filter has the same form as the Lee filter but with a different weighting function as shown in equation (5.4).

$$W = \frac{1 - \frac{C_u^2}{C_I^e}}{1 + C_c^2} \tag{5.4}$$

The Kuan filter made no approximation to the original model. From this point of view, it can be considered to be superior to the Lee Filter [38]. The Kuan filter can be derived directly by applying the MMSE criterion to the multiplicative model.

5.5 Frost Filter

The Frost filter differs from the Lee and Kuan filters with respect that the $\hat{x}_i$ is estimated by convolving the observed image with the impulse response of the coherent imaging system [37]. The system's impulse response is calculated by minimizing the MSE between the observed image and the estimated model, assumed to be an autoregressive process. The resulting filter after some simplifications can be written like equation (5.5).

$$\hat{x}_i = \exp\left(- KC_I^2 |t|\right) \tag{5.5}$$

where K is a constant controlling the damping rate of the impulse response function at the pixel to be filtered. When the variation coefficient C_I^2 is small, the filter behaves like an low pass filter smoothing out the speckles; when C_I^2 is large, it has a tendency to preserve the original observed image.

5.6 Double Filtering

Double filtering (DF) technique is a new image processing technique. It is not a work alone process. It simply forces to execute a selected non-wavelet denoising algorithm in two steps i.e. normal and reversed bases.

In first step selected filter are executed with original pixel values of image. This is a process done in all denoising process. In second step in addition, selected filter are executed again on pixel values which is reversed. The question is how these are reversed. It is a very simple process actually. Assume that image has 8-bit pixel value. Each pixel has value between "0" and "255".

$$xr_i = | x_i - 255 | \tag{5.6}$$

Where x_i is a specific value in noisy data set x and i is in the range from "0" to length of x. By this way if x_i is "0" than it becomes "255" and vice versa. Black becomes white and white becomes black. After denoising work on reversed xr data set, same process in (5.2) applied on xr to get x_i.

$$x_i = | xr_i - 255 | \tag{5.7}$$

The purpose of this technique is to enhance non-wavelet denoising schemes especially for adaptive median filtering. It is clear that standard deviations of different images are not expected to be the same. Adaptive median filter uses standard deviation of image so denoising response of this filter for the white and black speckle noise will not be the same. It will more likely to denoise white noise or black.

For evaluating the performance of double filtering, noise free and noise added images seen in Figure 5-2 are used. Initially PSNR evaluation is done between

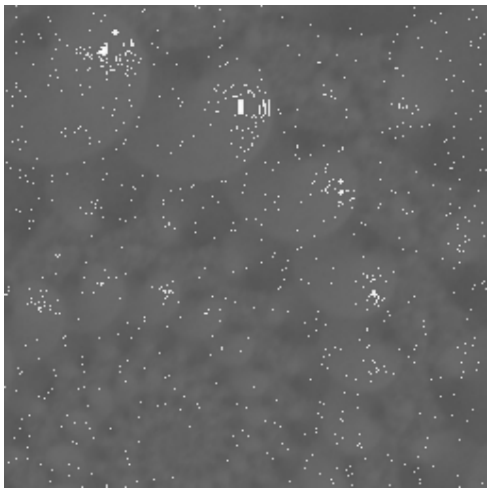
these two images and then all non-wavelet filters are executed on noisy image and resulted (denoised) image are compared with noise free image. It is seen from the Table 5.1 that DF technique gives better result in all cases.

Figure 5-2 Original Lena image and noisy Lena image with PSNR = 20.13

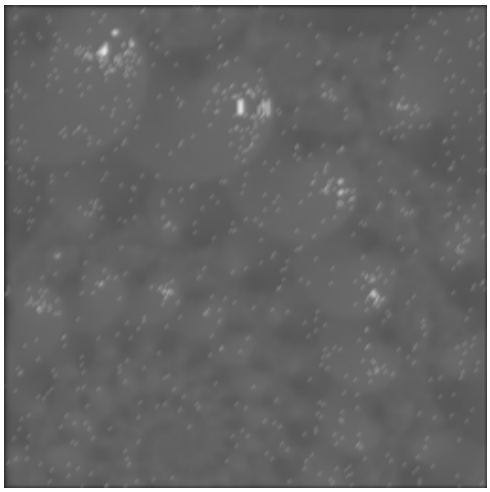
Table 5-1 Comparing single and double non-wavelet filtering schemes.

Filter type	Single	Double
	PSNR values	
Mean Filter	27,08	27,38
Median Filter	23.68	24.18
Adaptive Median Filter with σ_i = 0.7	23.78	24.20
Lee Filter	27,72	28,08
Kuan Filter	27,87	28,15
Frost Fılter	26,18	26,39

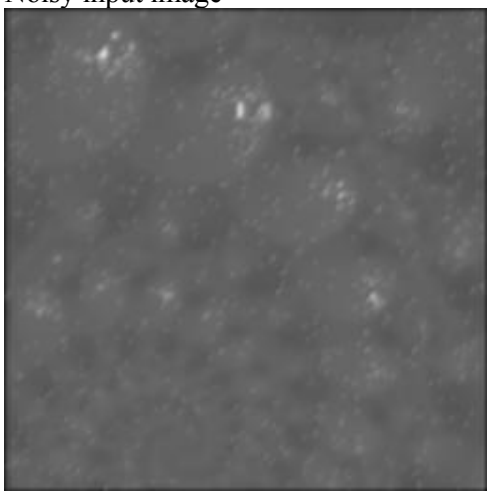
5.7 Results



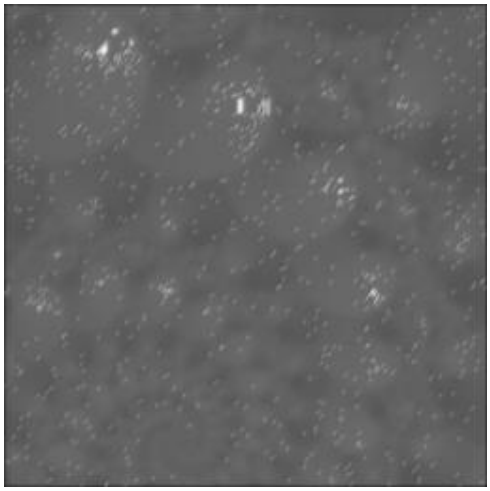
Noisy input image



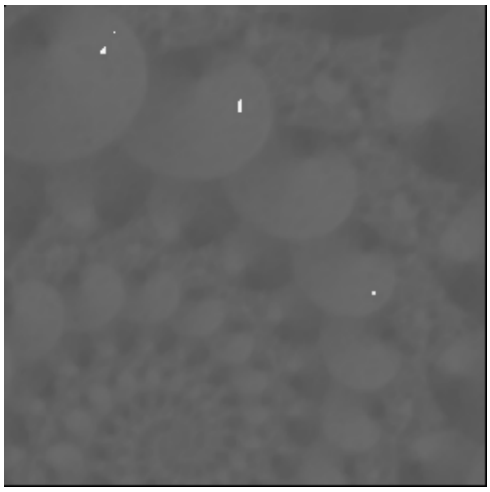
Mean Filter



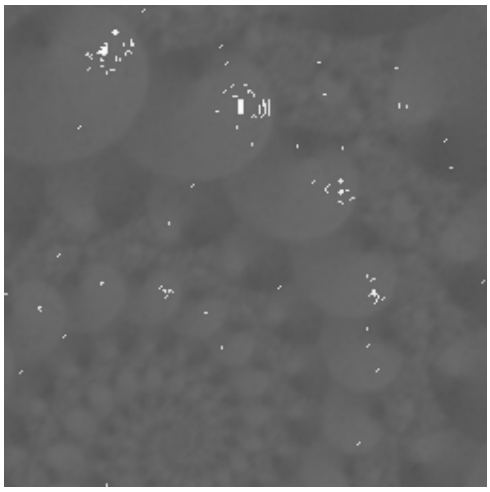
Kuan Filter



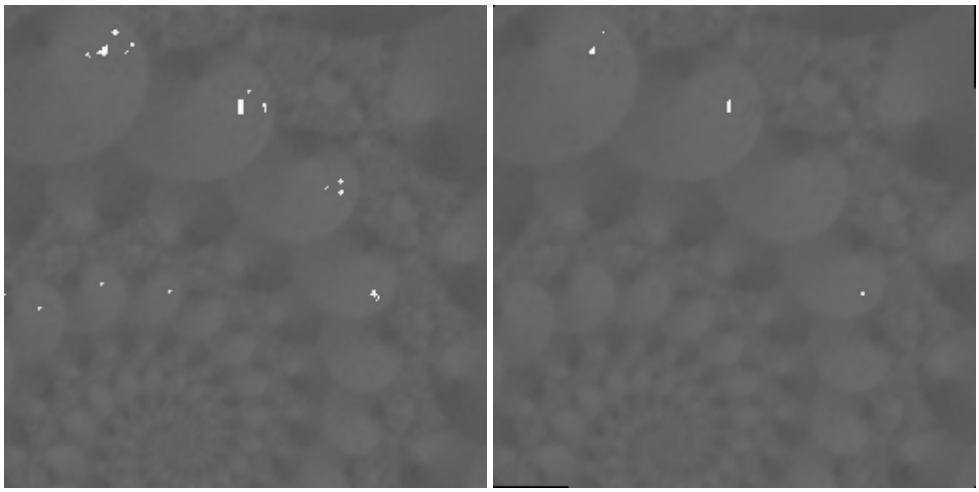
Frost Filter



Median filter



Adaptive Median Filter (σ_i =2)



e. Adaptive Median Filter (σ_i = 1.6) f. Adaptive Median Filter (σ_i =0.7)

Figure 5-3 Performances of non-wavelet filters on speckles

In Figure 5-3 Traditional Mean, Median and more statistically Kuan, Frost, and Adaptive Median filter with various settings are compared visually.

The image which has white speckles called “Noisy input image” in Figure 5-3 is taken account for this test. Speckle type of noise can be described as high intensive “high frequency” noise. In Figure 5-3 it is clearly seen that result of Mean filter based filters i.e. Mean, and Kuan looks more smoothed. Frost filter, a bit more sharper then Mean and Kuan, give similar result. Result of median filter seen in same Figure looks very good. All of the speckles except for very large ones are removed successfully without significant smoothing. For the adaptive case of Median filter, it is seen that while σ_i is getting smaller, result is approaching to standard median filter’s result. In the same Figure again image filtered by Adaptive Median with σ_i =0.7 looks like to image filtered by Median.

5.8 Conclusion

In this chapter some basic and more advanced non-wavelet denoising algorithms were studied.

In Mean filter all pixels are replaced with mean value of neighborhoods. This filter have very good result shown in Table 5-1 but it tends to smooth images more then other filters (see Figure 5-3). Other more advanced filters Frost and especially mean based Kuan filter seem to be slightly better (see Table 5-1 and Figure 5.3).

Median based filters have preserve image detail more than mean filter but have pure denoising capability especially for images with various type of noise when executed in single mode (see Table 5-1). However in Figure 5-3 it is seen that this filter is very usable for high intensive noises for example some resident noises after wavelet denoising. Median filters are more likely to be an after filter. Main advantage of Adaptive Median filter lies on its input parameter which brings possibility to change intensity of threshold for specific conditions. For some image types especially for astronomical images stellar objects can be interpreted as a sort of impulsive noise in deep-sky images, especially when they are numerous: they are more or less randomly distributed, they are bright, and they are small. Taking advantage of this fact, the adaptive median filter can be used to remove or isolate stars on images while decreasing noise.

Finally DF technique was introduced in this chapter. DF is not a process which works for all times perfectly but is needed due to weak response of adaptive median filter in some cases. Actually adaptive median filter is very good for denoising low or high intensity speckles. Its denoising intensity can be adjusted using input parameter. However due to its nature it hasn't got same effect on both white and black noise. By means of DF black noises are eliminated after white noises or vice versa.

As a result, two options seem to be better then others. These are Adaptive Median and Frost filters. These two filters which have different algorithms can be used for different purposes. Further queries will be done on Chapter 7.

CHAPTER 6

TOOLBOX

Wavelet Denoising Toolbox (WDT) version 1.0 was implemented over the ImageJ computing environment with graphical interface for developing wavelet and non-wavelet based algorithms for the analysis and denoising images [31].

ImageJ mentioned in Chapter 2 is a platform free pure multithreaded Java image processing program with many functions presented. According to its creator, It is the world's fastest pure Java image processing program. It includes all basic processes for the images such as reading, displaying, editing, saving and printing. It includes also many mathematical well known image process under the Process tab on the main interface window seen in Figure 6-1. *Analyze* tab in the same Figure also contains some basic analyses tools.

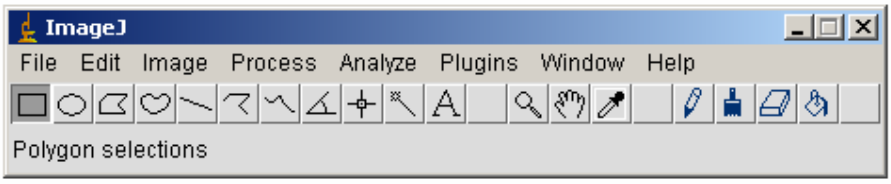


Figure 6-1 Main interface of ImageJ

WDT was developed using ImageJ's plugin extension. Because of the nature of ImageJ, WDT operates with pure java code. Most important advantages of Java language such as object oriented feature, automatic memory management and of course its libraries were used.

Images must be opened before starting WDT. Opening the image (see Figure 6-2) is relatively easy by pressing *Open* option in *File* menu and selecting image from the file browser appeared on the screen.

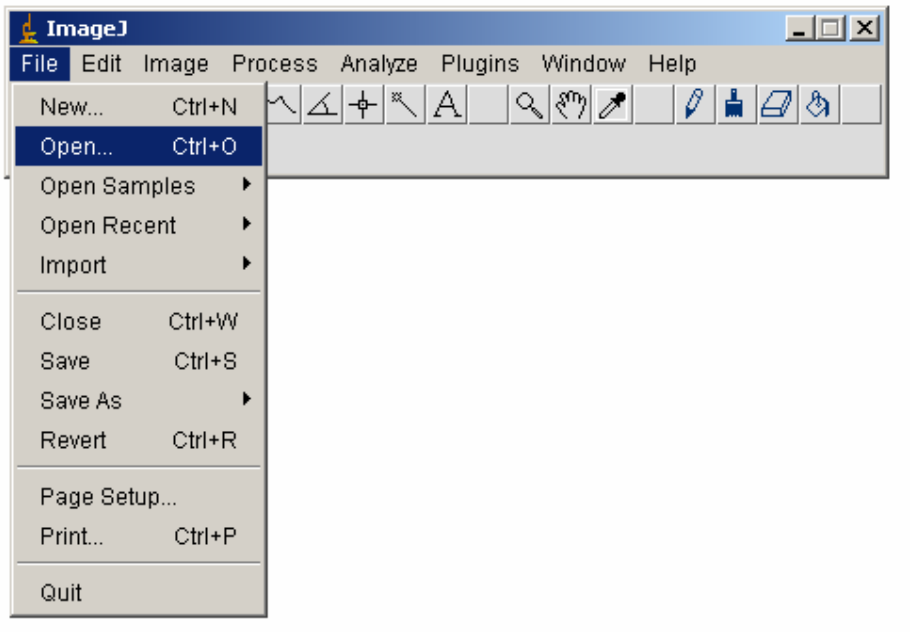


Figure 6-2 Opening image in ImageJ

WDT is capable of loading only one image at the same time. If more than one opened image available only active "selected" one is loaded. Batch process over the images is not possible.

In order to start WDT, *Image Denoising* option of the *Wavelet* menu which is available under the *Plugins* tag must be selected (see Figure 6-3). Image opened is automatically loaded while starting of WDT. At the same time this image is analyzed automatically to check it is suitable or not prior to action. All images selected for WDT must have 8-bit data value. Color images with larger scales above 8-bits are not allowed. They must be converted to 8-bit scale in order to operate. This can be done using ImageJ's own image conversion option called *Types* available under the *Image* tag. Physical size "height & width" of selected images is another issue that must be criticized. Due to limitations of ImageJ, WDT

can not handle images with larger than “2048 & 2048”. Images equal or under this size are accepted but another criticism must be done at this point. Due to nature of wavelet transformation width & height of the image must be equal to each other. However this is not a problem for WDT. Images are converted automatically to suitable size if needed.

Figure 6-3 Starting wavelet denoising plugin of ImageJ

In Figure 6-4 main interface of WDT screen are seen. *SURE Shrinking* process which use *CDF 2-4* wavelet base is executed before the *Adaptive Median Filtering* with σ_i equals to "2" and *Double Filtering* enabled.

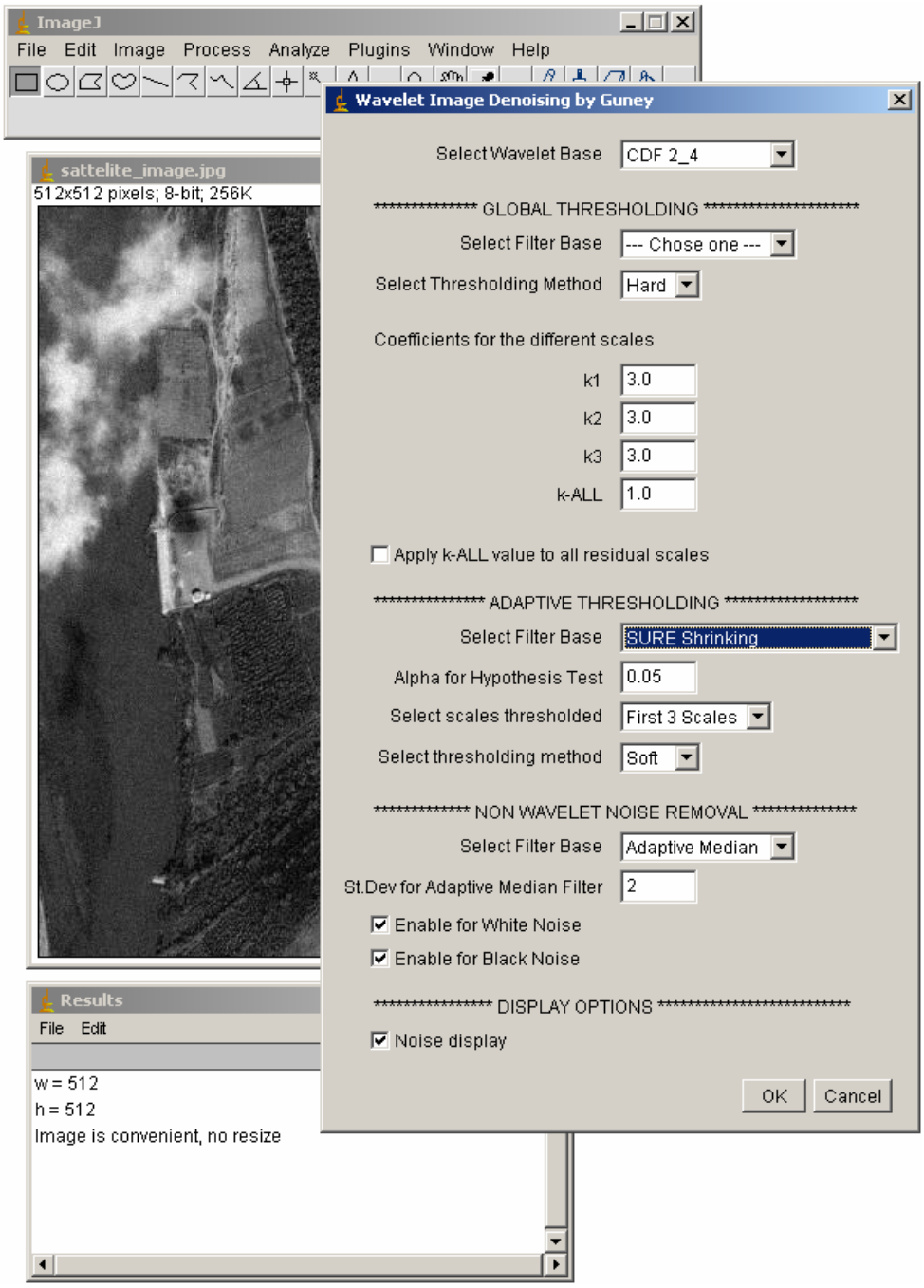


Figure 6-4 Selecting denoising parameters

Interface of WDT consists of three sections. Among them *Global thresholding*, and *Adaptive Thresholding* include wavelet based methods other called *Non-wavelet Noise Removal* includes traditional non-wavelet denoising methods.

At the top of the interface window one of the wavelet base mentioned in Chapter 3 can be selected for all wavelet based methods i.e *Globa Thresholding* and *Adaptive Thresholding* processes.

Global thresholding processes including *VisuShrink* and *Wiener Filter* mentioned in Chapter 4 operate with user defined scale coefficents. These are *K1, K2, K3* and *K-ALL* called first, second, third and forth scale coefficents respectively. If needed input value of *K-ALL* can be used for all resudual scales by simply checking the relevant box. Note that although *Wiener Filter* needs pre-defined parameter for each scale it is not a global thresholding procedure exactly. But surely it is not a automatic procedure like ones listed in adaptive thresholding section also.

Adaptive thresholding section includes all processes mentioned in Section 4.2. In order to operate these processes except for *Bayesian Shrinking* thresholding scales and thresholding method must be selected before. Note that *Bayesian Shrinking* was implemented as a fully automatic procedure. *Alpha* used for *Hypothesis Test* is another pre-defined parameter here.

Non-wavelet noise removal processes mentioned in Chapter 5 are found below the interface of WDT. All pre-defined parameters in this section are used for *Adaptive Median* process. Double filtering process mentioned in Section 5.3 can be activated simply by checking the box at the left side of *Enable Black Noise* tag while *Enable White Noise* tag is checked. The adaptive median process will not work if neither of them *Enable Black Noise* or *Enable White Noise* are selected.

All of the processes mentioned above can be operated in the specified order at the same attempt. Adaptive thresholding process which have the highest priority among them are in the first in queue, global thresholding processes are in the middle and non-wavelet denoising processes are at the end.

Figure 6-5 Result; noisy input image, removed noise, and denoised image with summary window

In the result screen (see Figure 6-5) noisy input image, denoised image, and noises which were subtracted from noisy image are seen. Result window contains summary of operation with RMSE and PSNR values. Note that these two values simply indicate the difference between noisy input image and final denoised image. They can not be used as a performance indicator for processes.

CHAPTER 7

TESTS AND CONCLUSION

In previous chapters wavelet models as well as denoising algorithms were tested on demo images. Some information about performances of implemented algorithms were found out. However the question “Can these algorithms show different performances on different images ?” is still unknown.

In this chapter combinations of wavelet models and denoising algorithms are tested on various images. For this purpose, in the first step, all denoising methods using same wavelet model are tested on artificially noise added versions of noise free images. The wavelet model used here is Coiflet-6 one of the most successful wavelet models implemented in this work (see Section 3.9).

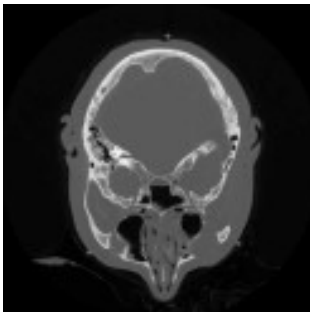
The second step is totally about SAR imagery. For this purpose different portions of an original SAR image are used. Note that SAR image used here is a raw type. It includes noise originally and no denoising modification was applied before. Best denoising combination(s) plus best wavelet model(s) are found out for SAR imagery.

For the first step; in Figure 7-1 and Figure 7-2, noise free “reference” test images and noise added versions are shown respectively. Amount of Gaussian noise so called white noise added to each image and PSNR evaluation of noise free and noise added images are shown also in Figure 7-2. Speckle type of noise so called Salt&Peppers (SP) along with Gaussian noise is also added some images seen Figure 7-1 (d,g,h). The results based on PSNR value are shown on Table 7-1 and Table 7-2.

a. old movie

b. historical

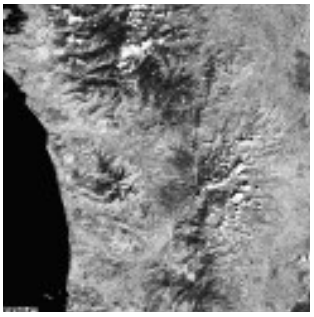
c. Lena



d. MRI

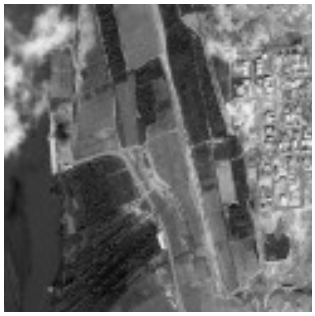
e. peppers

f. planet surface



g. satellite image

h. moon



i. satellite image

Figure 7-1 Noise free image set for final test.

a. noisy old movie

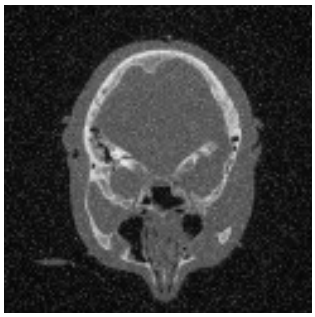
σ_g = 15, PSNR = 24.6

b. noisy historical

σ_g = 20, PSNR = 21.64

c. noisy Lena

σ_g = 20, PSNR = 20.22



d. noisy MRI with SP

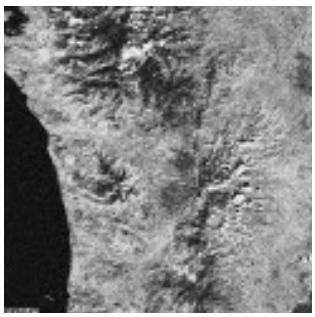
σ_g = 15, PSNR = 16.32

e. noisy peppers

σ_g = 30, PSNR = 18.40

f. noisy planet surface

σ_g = 20, PSNR = 21.46

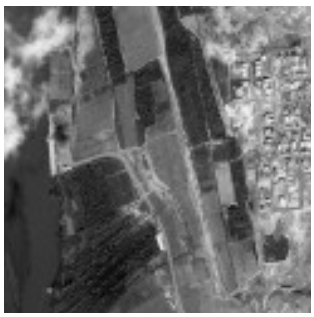


g. noisy satellite image with SP

σ_g = 25, PSNR = 20.49

h. noisy moon with SP

σ_g = 20,PSNR = 16.18



i. noisy satellite city image

σ_g = 15, PSNR = 28.73

Figure 7-2 Noisy image set for final test with additive several Gaussian noise with $N(0, \sigma^2)$ and Salt&Peppers (SP).

Table 7-1 Performance evaluations of wavelet filters (1)

İmage	Noisy	Mean	Visu hard, first 3 scale T_m =(2,1,0.5)	Wiener hard, first 3 scale T_m =(3,2,1)
a. old movie	24.6	27.08	26.15	27.10
b. historical	21.64	20.47	20.05 20.03 (F)	21.23 20.90
c. Lena	20.22	27.68	27.62 27.85 (F)	26.95 27.64 (F)
d. MRI (SP)	16.32	23.65	22.36 24.10 (F)	18.27 20.18 (F)
e. peppers	18.40	26.27	25.91 26.26 (F)	23.83 24.76 (F)
f. planet surface	21.46	20.59	20.01 19.90	21.50 21.40 (F)
g. satellite image (SP)	15.91	16.40	16.15 16.25 (F)	15.98 16.74 (F)
h. moon (SP)	16.18	24.32	22.85 25.07 (F)	18.09 26.81 (F)
i. low noise sat. city image	28.73	25.85	26.92 28.10 (F)	28.77 28.94(F)

Table 7-2 Performance evaluations of wavelet filters (2)

Image	Noisy	Sure soft, first 3 scale	Bayes soft	GCV soft, first-2 scale	Hypothesis soft, first-3 scale
a. old movie	24.6	28.31 **28.64** (F)	26.22 27.09 (F)	27.53 27.47 (F)	27.77, α=0.05 28.59, α=0.5 **28.59** (F)
b. historical	21.64	21.88 21.95 (F)	21.77 **22.40** (F)	20.36 20.09 (F)	20.94,α=0.05 22.13,α=0.5 21.92 (F)
c. Lena	20.22	23.33 27.60 (F)	22.36 27.87 (F)	27.53 28.05 (F)	27.65,α=0.05 **28.14** (F)
d. MRI (SP)	16.32	17.12 28.20 (F)	16.46 **29.65** (F)	17.03 **29.53** (F)	18.39, α=0.5 18.91, α=0.05 27.57 (F)
e. peppers	18.40	21.00 26.12 (F)	19.64 25.56 (F)	26.11 26.45 (F)	23.48,α=0.5 24.93,α=0.05 **26.50 (F)**
f. planet surface	21.46	23.28 **23.35** (F)	21.87 22.60 (F)	22.04 22.10 (F)	22.70 (F) 23.03, α=0.05 23.78, α=0.5
g. satellite image (SP)	15.91	16.96 **18.76** (F)	16.89 **18.70** (F)	16.55 17.30 (F)	17.00, α=0.5 17.15, α=0.05 18.16 (F)
h. moon (SP)	16.18	18.14 31.01 (F)	16.3 **31.77** (F)	16.55 **31.90** (F)	18.04, α=0.5 18.47, α=0.05 30.40 (F)
i. sat. city image	28.73	28.88 28.13 (F)	**30.11** 30.10 (F)	26.85 26.94 (F)	25.77, α=0.05 27.08, α=0.5 27.47 (F)

From Table 7-1 and Table 7-2 it is seen that data adaptive wavelet shrinkage methods are clearly winner. Also between all of these methods there is no one which is the best for all cases. However the idea about the best shrinkage method for specific image can be found out.

In the test above "*F"* character are shown with some results. This means that one of the non-wavelet denoising algorithms was applied along with wavelet denoising scheme. Here adaptive median filter with same parameter were used for all images. Results show that performance values of wavelet filters changed dramatically with the aid of non-wavelet methods.

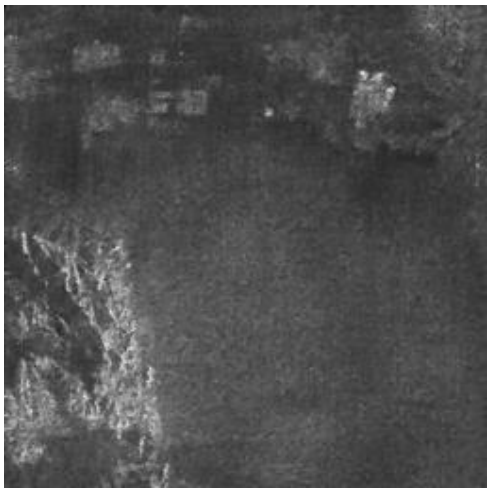
The images used in this test can be classified into two categories. First is more detailed images such as Satellite images, Planet surface and Historical. Second is ordinary images i.e. old movie, Lena, Peppers, Moon and M.R.I. For more detailed images Bayesian and Sure denoising methods works well. With combination of non-wavelet denoising algorithm, Bayesian filter has very good results. The reason can be explained that Bayesian filter has own shrinkage method using neither soft nor hard thresholding. Other methods using soft or hard thresholding schemes calculate a single threshold value for all coefficients in specific wavelet scale. Remember that Bayesian filter estimate each wavelet coefficients using distributions of prior coefficient. This means private threshold values for each wavelet coefficient. This feature is very helpful to obtain sensitive results for very detailed images. However for images including high intensive speckle type noise, it is seen that this feature fails. On the other hand combination of adaptive median filter with Bayesian filter interestingly shows highest performance values for images with SP noise. The explanation is simple. Bayesian filter heavily shrinks small-in-magnitude coefficients. On the other hand adaptive median filter is very good for denoising large magnitude arguments when small-in-magnitude arguments are low in number. Bayesian method gives perfect work area to adaptive median filter. These results can be very attractive for denoising SAR images. Note that SAR images contain speckle noise as well as other noise [39].

For the second type “ordinary images”, Sure shrinking and especially Hypothesis test work very well. Ordinary images used in this test i.e. Peppers, Lena, and M.R.I are not very detailed images. In each scale a single threshold value calculated by Hypothesis test and Sure works well for shrinking not only small-in-magnitude arguments also larger ones. These methods give a bit smoother images then Bayesian filter but this is not important for low detailed images.

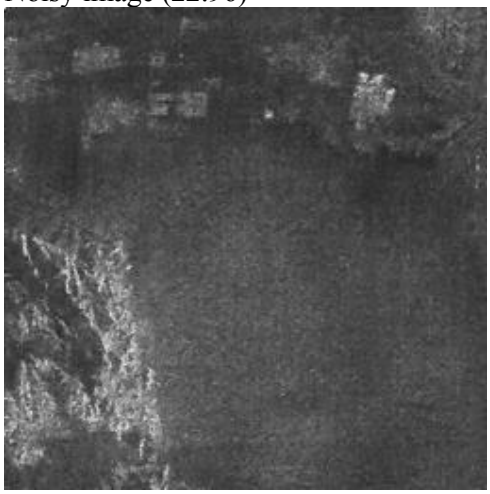
For the second step; three different portions of SAR image are used. It can be assumed that these SAR images contain speckle type of noise as well as other noise. Remember that almost all of the white noises of a noisy image are resident on first three wavelet scale, mostly on the first scale. In many situation, denoising on first three wavelet scales can be enough for removing white noise. However for speckles, different methods must be considered. Because of the high intensity of these speckles, denoising with wavelet data adaptive methods were executed for all wavelet scales except for highest order. Remember that high intensive characteristics of image are available on higher order scales. Also classical non-wavelet algorithms which are well known for speckle denoising are used after denoising on wavelet domain to achieve further success. In figure 7.3, 7.4 and 7.5 noisy image and denoised versions are shown. All denoising schemes use Coiflet-6 wavelet model. Because of the very weak performance of universal thresholding scheme (see Section 4.6) VisuShrink are not used here.

Performance evaluation of denoised SAR images is done by comparing Mean and STD of raw and denoised images and also visually. Remember that for the PSNR evaluation of a denoised image a reference noise free image is also used (see Section 3.8). In real world applications like SAR imagery, no reference images available so different methods must be taken account.

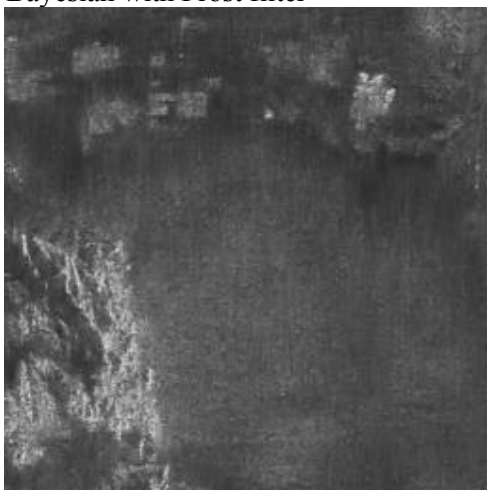
Noisy image (22.96)



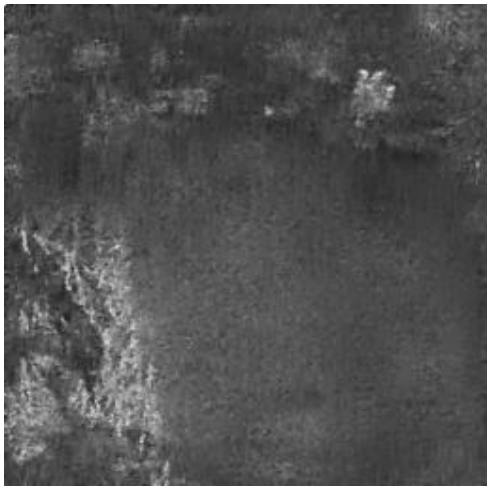
Bayesian with Frost filter



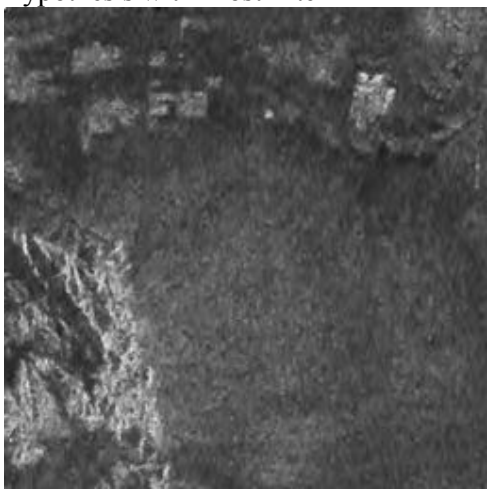
SURE with Frost filter



Hypothesis with Frost filter

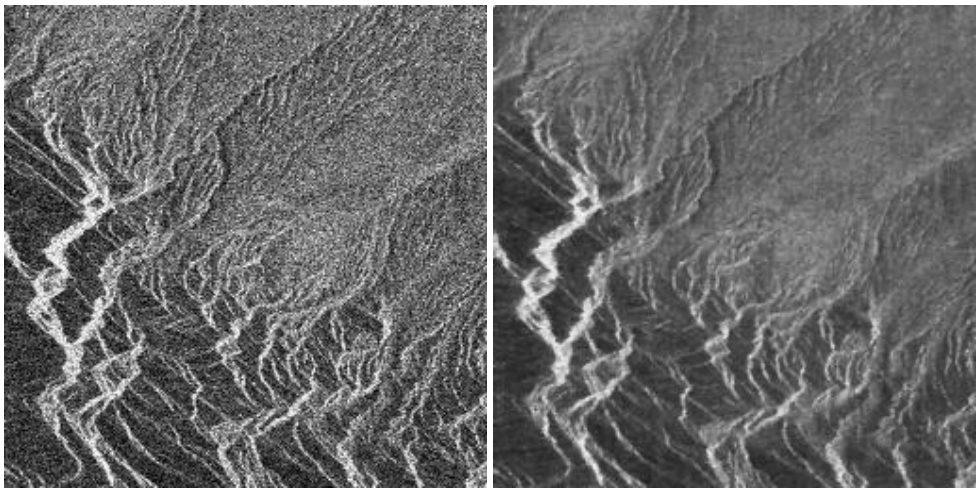


GCV with Frost filter

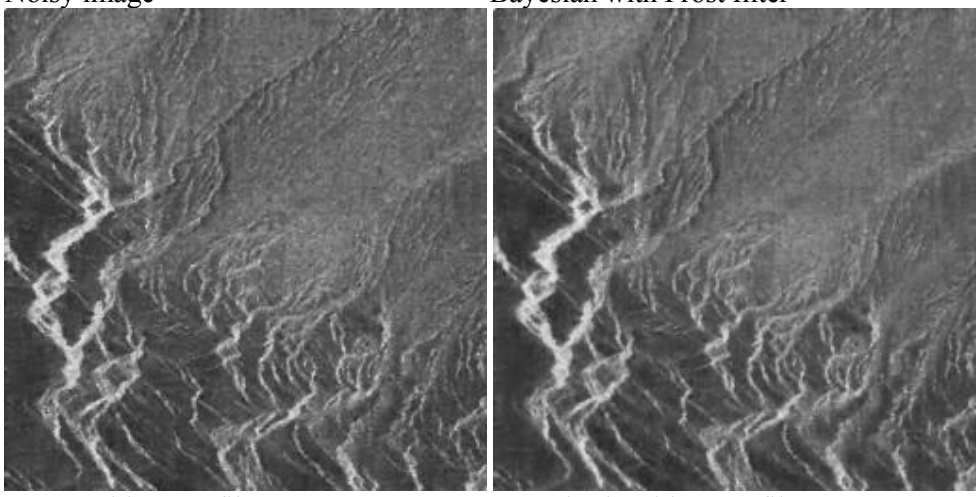


Wiener with Frost filter

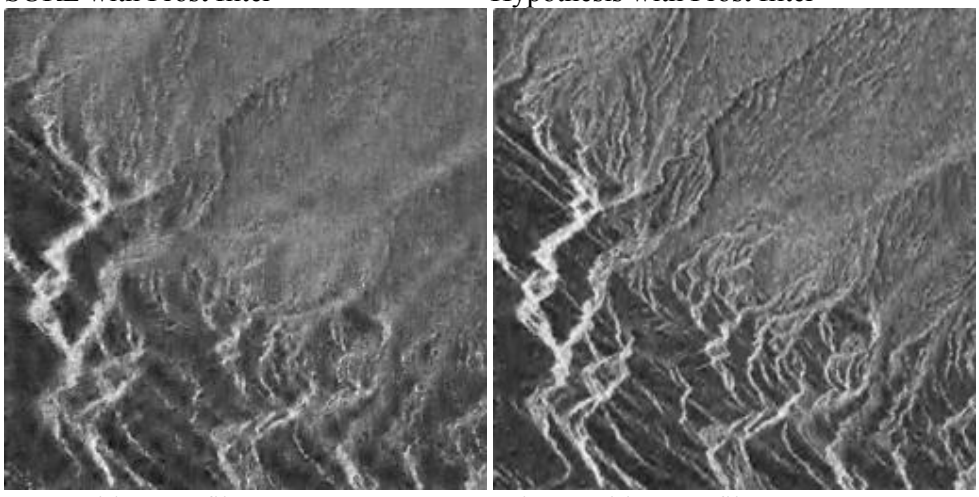
Figure 7-3 Denoising of a SAR image-1



Noisy image | Bayesian with Frost filter

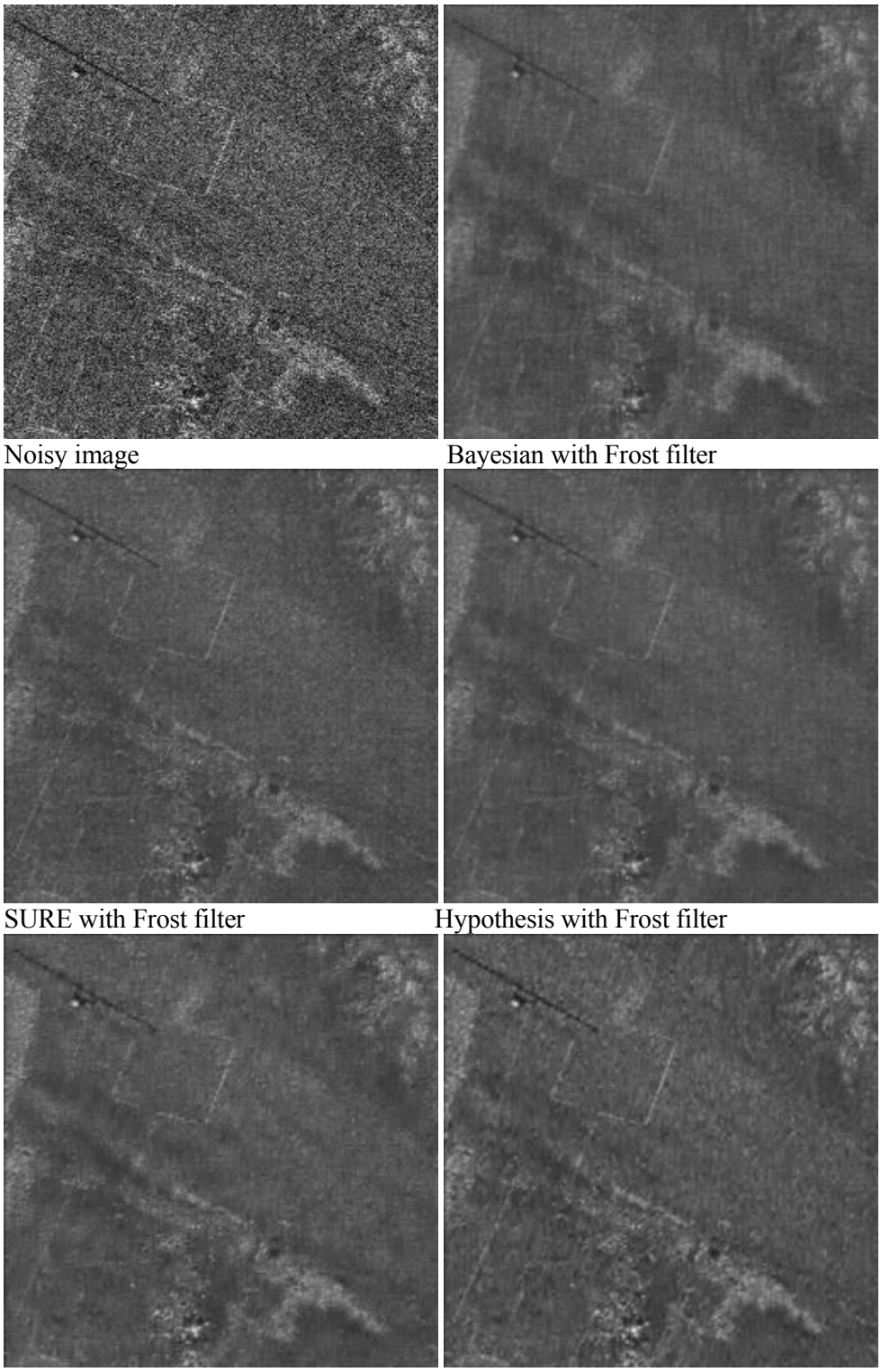


SURE with Frost filter | Hypothesis with Frost filter



GCV with Frost filter | Wiener with Frost filter

Figure 7-4 Denoising of a SAR image 2



Noisy image | Bayesian with Frost filter

SURE with Frost filter | Hypothesis with Frost filter

GCV with Frost filter | Wiener with Frost filter

Figure 7-5 Denoising of a SAR image 3

Table 7-3 Standard deviation of SAR images with and without (NA) denoising

Method	Image-1	Image-2	Image-3
	Mean - STD		
Sure + Frost	82.47 - 6.86	104.47 - 9.01	85.21 - 9.70
Bayes + Frost	82.47 - 9.73	104.47 - 11.36	85.21 - 9.95
Hypothesis + Frost	82.47 - 10.19	104.47 - 12.40	85.21 - 10.43
GCV + Frost	82.47 - 10.55	104.47 - 13.45	85.21 - 10.97
Wiener + Frost	82.47 - 10.90	104.47 - 13.47	85.21 - 11.24
GCV	82.72 - 12.23	104.78 - 15.40	85.50 - 12.37
Hypothesis	82.72 - 12.7	104.78 - 15.94	85.50 - 13.37
Wiener	82.72 - 13.6	104.78 - 16.45	85.50 - 13.75
Bayes	82.72 - 14.7	104.78 - 18.29	85.50 - 13.79
Sure	82.72 - 15.7	104.78 - 19.31	85.50 - 16.65
NA (Noisy Image)	82.72 - 22.96	104.78 - 27.58	85.50 - 24.20

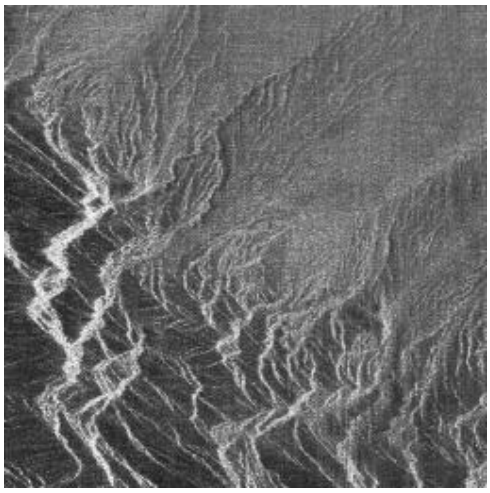
In Table 7-3 Mean and STD of original SAR images (NA) and denoised SAR images seen in Figure 7-3, 7-4, and 7-5 are shown. For the purpose of evaluating the performance of the filters on SAR images, two quantities of Mean and STD are used. Based on these two quantities, the best performance filter is selected if the Mean of filtered image is close to the original image while the STD of filtered image has the minimum value [40].

Acording to the results shown in Table 7-3 it is clearly seen that for all of three portion of a SAR image, wavelet based denoising methods with combination of non-wavelet method give best results. Note that in each method, Mean values have very minor differences not accounted. Only significant differences are occurred when Frost filter is on. But this seems to be not important also. The question is “Why does Wiener makes better then Bayes or Sure without using non-wavelet methods?”. The Answer is simple; Wiener filter is not a data adaptive method and also it is using a predefined threshold multiplier. It can be said that the

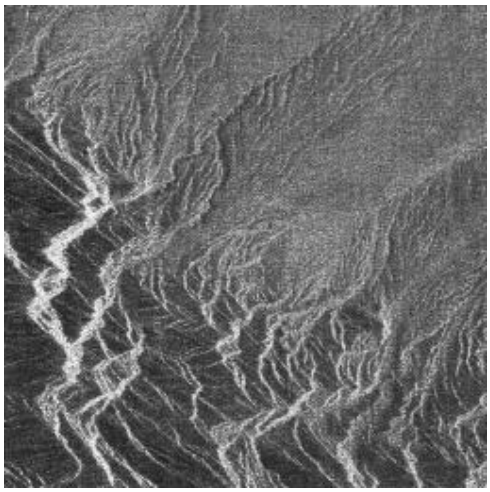
threshold value estimated by Wiener is not expected to be as sensitive as values of Bayes or Sure. Remember that SAR images include speckles. This sensitiveness sometimes makes better job like thresholding some of the speckles on images then Bayes or Sure can do alone. But it is seen that by the help of non-wavelet methods, Bayesian and especially Sure schemes are clearly winner. For SAR images non-wavelet methods seem a must for successful denoising. Other methods; i.e. GCV yields more smoothed results and Wiener filter has weak denoising potential. Performance of Hypothesis testing is on average. Although the results of Hypothesis test and Bayesian are very close to each other, Bayesian seem to be slightly more detailed with respect to results of Hypothesis testing.

After this point tests will continue using Sure and Bayesian wavelet denoising schemes with different wavelet models.

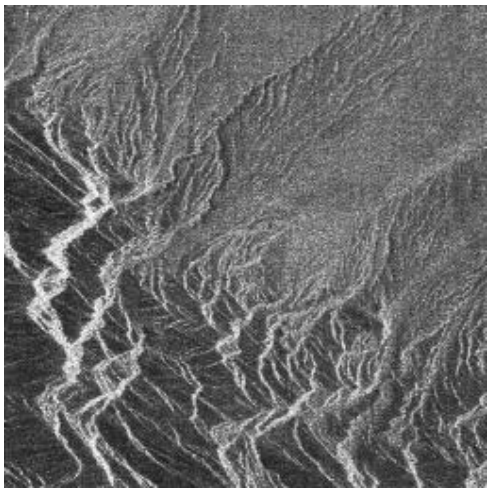
Noisy Image



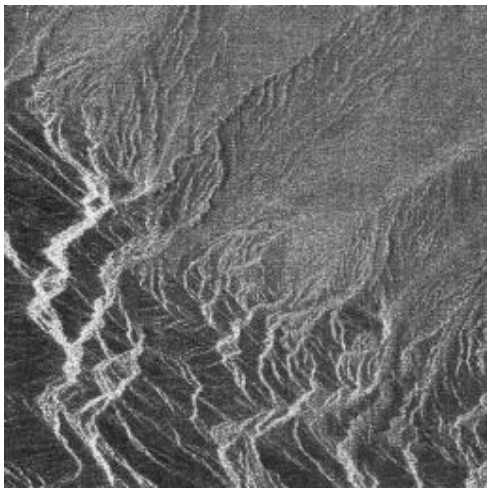
Bayes with CDF (2-4)



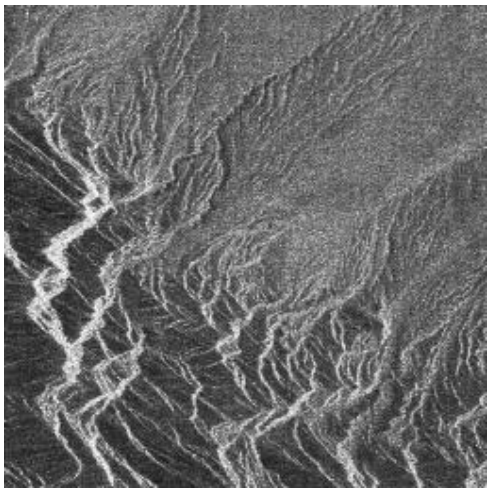
Bayes with P. Coiflet 4



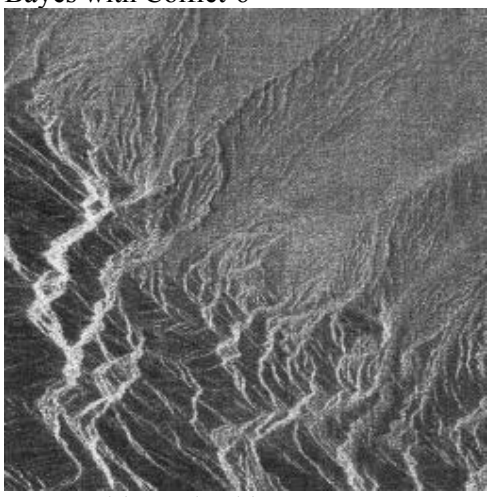
Bayes with Battle Lemaire



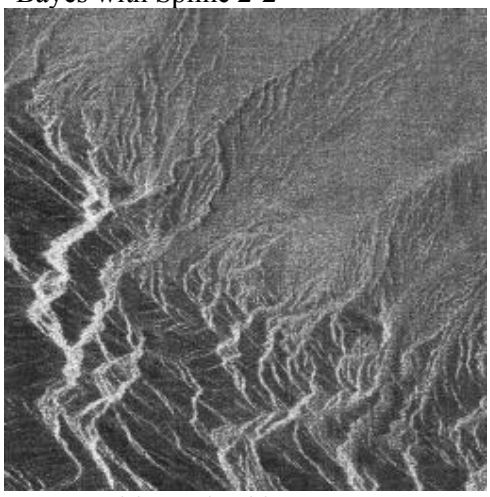
Bayes with Coiflet-6



Bayes with Spline 2-2



Bayes with Daubechie-20



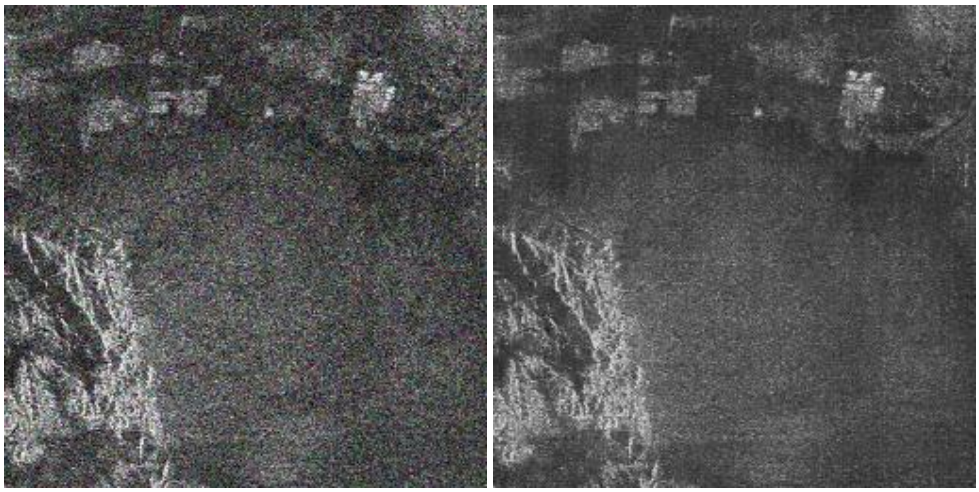
Bayes with Daubechie-4

Figure 7-6 Denoising SAR image-2 using Bayes scheme with different wavelet models.

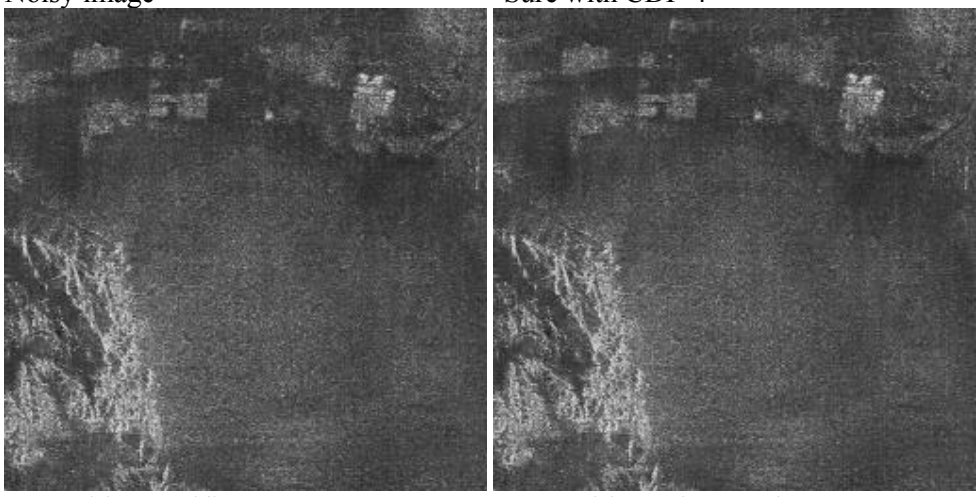
Table 7-4 Standard deviation of SAR image-2 with and without Bayes scheme using various wavelet models

Method	Image 2 Mean - STD
Bayes with Daubechie 20	104.78 - 18.05
Bayes with Daubechie 12	104.78 - 18.15
Bayes with Daubechie 4	104.78 - 18.23
Bayes with Coiflet 6	104.78 - 18.29
Bayes with Coiflet 4	104.78 - 18.29
Bayes with Coiflet 2	104.78 - 18.31
Bayes with Battle Lemaire	104.78 - 18.33
Bayes with P.Coiflet	104.78 - 19.71
Bayes with CDF 2-4	104.78 - 19.90
Bayes with Spline 2-2	104.78 - 20.12
NA	104.78 - 27.58

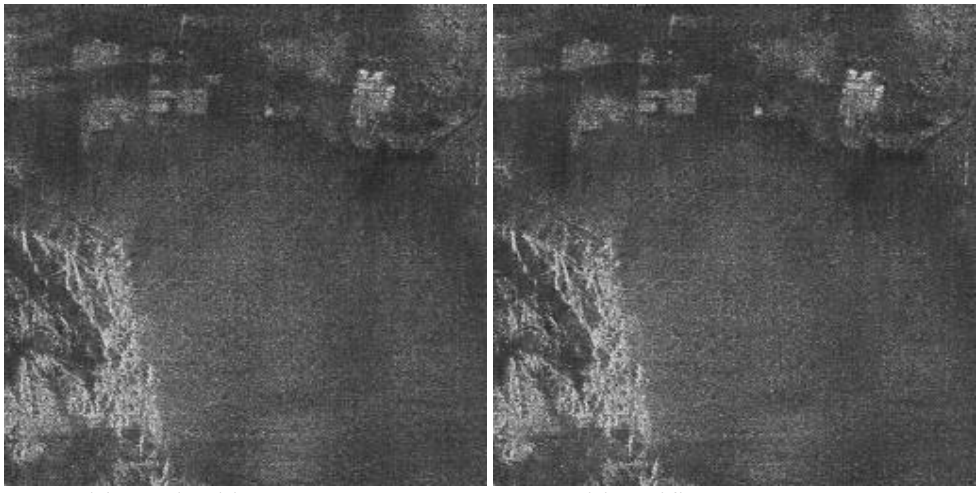
In Table 7-4 STD of SAR image-2 seen in Figure 7-6 with and without applying Bayes schemes using various wavelet models are shown. Note that here not all of the wavelet models were used. Accordingly wavelet performance results shown in Section 3.8, wavelet models which have higher scores are used. In order to see performances of same wavelet model with low and high order, Daubechies and Coifflets with different orders were also used. It is seen that the highest order Daubechie works better than lower ones also same situation goes for Coiflets in this test. Because of the lower distortion capability of long filters, they have better denoising feature than short ones. Daubechies, Coiflets and Battle Lemaire wavelet models also have very good and similar results.



Noisy image | Sure with CDF-4



Sure with P. Coiflet 4 | Sure with Battle Lemaire



Sure with Daubechie-20 | Sure with Coiflet-6

Figure 7-7 Denoising SAR image-1 using Sure scheme with different wavelet models

Table 7-5 Standard deviation of SAR image-2 with and without Sure scheme using various wavelet models

Method	Image 1 Mean - STD
Sure with Daubechie 20	82.72 - 15.60
Sure with Daubechie 12	82.72 - 15.69
Sure with Coiflet 6	82.72 - 15.72
Sure with Coiflet 4	82.72 - 15.73
Sure with Coiflet 2	82.72 - 15.76
Sure with Battle Lemaire	82.72 - 15.77
Sure with Daubechie 4	82.72 - 15.81
Sure with P.Coiflet	82.72 - 15.91
Sure with Spline 2-2	82.72 - 15.97
Sure with CDF 2-4	82.72 - 16.45
NA	82.72 - 22.95

In the test shown in Figure 7-7 and Table 7-5 various wavelet models are used with Sure denoising scheme. Like previous test with Bayes, Daubechies and Coifflets with different orders were also used here. Although most of the wavelet models have very similar results, the best performers seem to be Daubechies with higher orders. In both cases, Sure and Bayes, higher order Daubechies and Coiflets give best overall results among all wavelet models.

After these tests, the last step is to find best non-wavelet denoising scheme over these methods. Between all mean based filters i.e : Mean, Lee and Kuan, only Kuan is used in this test due to its superior results compared to others. Median and more statistically Frost and Adaptive Median filter are also used in this test.

In Table 7-6 and Figure 7-8 results are shown. It is seen that Frost filter is the most successful one. Although STD value of Median filter is lower, Mean value has noticeable difference. Frost filter mathematically seems to best one. However

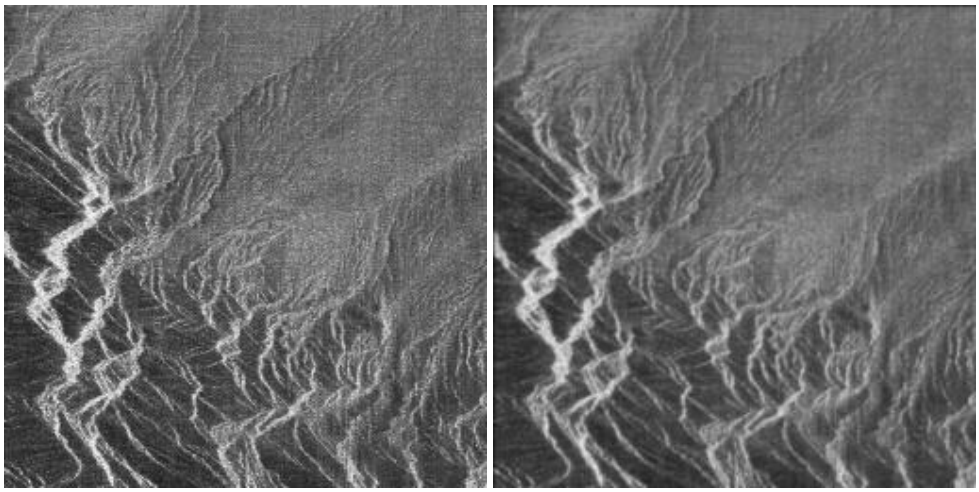
when the results are examined visually (see Figure 7-8) Kuan, Median, and also Frost filters yield more smoothed results with respect to adaptive median filter. Note that in this case also Frost filter gives better result over Kuan and Median filter.

In image denoising works sharpness is much more valuable then smothness in some situation. When any detail lost is unacceptable, adaptive median filter with appropriate parameter can be chosen. On the other hand if smoothness is more valuable then sharpness, Frost filter seems to be best one between all of the non-wavelet based filters in this work.

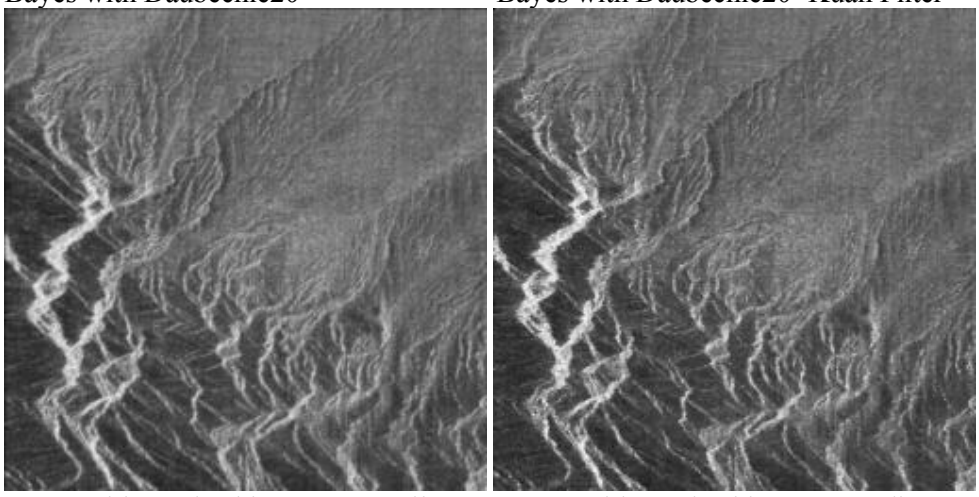
Table 7-6 Standard deviation of SAR image-2 with Bayes denoising scheme + Daubechie 20 and with and without non-wavelet filters.

Method	Image 2 Mean - STD
Bayes with Daubechie 20 + Median	95.74 - 8.19
Bayes with Daubechie 20 + Frost	104.47 - 11.05
Bayes with Daubechie 20 + Kuan	104.39 - 11.19
Bayes with Daubechie 20 + A.Median (σ_i=1.2) + DF	100.63 - 12.01
Bayes with Daubechie 20 + A.Median (σ_i=1.2)	101.09 - 12.55
Bayes with Daubechie 20	104.78 - 18.05
NA	104.78 - 27.58

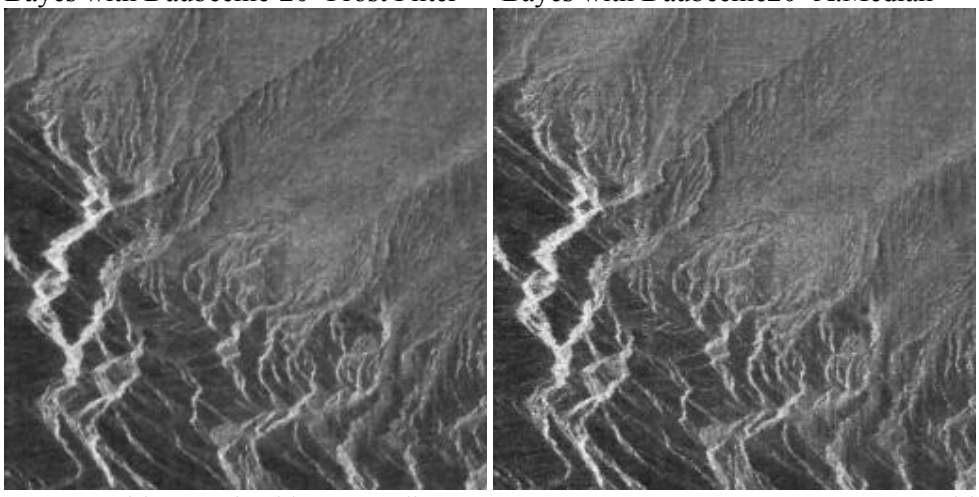
The Mean and STD values of Bayes plus non-wavelet methods are shown in Table 7-6. The images are also shown in Figure 7-8.



Bayes with Daubechie20 Bayes with Daubechie20+Kuan Filter



Bayes with Daubechie-20+Frost Filter Bayes with Daubechie20+A.Median



Bayes with Daubechie20+Median Bayes with Daubechie20+A.Median+DF

Figure 7-8 Denoising SAR image-2 using Bayes+D20 and non-wavelet models

REFERENCES

[1] Riol, O., and Vetterli. M., Wavelets and signal processing, IEEE Signal Processing Magazine. Vol. 8, No. 4, pp. 14-38, 1991.

[2] Mallat, S., A theory for multiresolution signal decomposition: the wavelet representation, IEEE Transactions on Pattern Recognition and Machine Intelligence. Vol. 11, No. 7, pp. 674-693, 1989.

[3] Daubechies, I., The wavelet transform, time-frequency localization and signal analysis, IEEE Transactions on Information Theory. Vol. 36, No. 5, pp. 961-1005, 1990.

[4] Vetterli, M., and Kovacevic J., Wavelets and subband coding, Prentice Hall, Englewood Cliffs, NJ, USA, 1995.

[5] Mallat, S., A wavelet tour of signal processing, Second addition, Academic Press, 1998.

[6] Daubechies, I., Ten Lectures on Wavelets, SIAM, Pennsylvania, 1992.

[7] Poularikas, A. D., The Transforms and Applications Handbook, CRC/IEEE Press, pp. 747-827, 2000

[8] Reza, Ali M., Wavelet Characteristics, Spire Lab, UWM, 1999.

[9] Unser, M., Wavelets Applications in Signal and Image Processing V, SPIE Vol. 3169, pp. 422-431, 1997.

[10] Valens, C., A really friendly guide to wavelet, "http://perso.wanadoo.fr/polyvalens/clemens/download/arfgtw.pdf", last access date: 05.23.2006

[11] Burt, P., and Adelson, E., The Laplacian pyramid as a compact image code, IEEE Trans.Comm., COM-31, 482-540, 1983.

[12] Cohen, A., Daubechies, I., and Faveau, J.-C., Biorthogonal Bases of Compactly Supported Wavelets, Communications on Pure and Applied Mathematics, Vol. XLV, pp. 485-560, 1992.

[13] Battle, G., A block spin construction of ondelettes. Part I: Lemarié functions, Commun. Math. Phys., Vol. 110, pp. 601-615, 1987.

[14] Lemarié, P. G., Ondelettes à localisation exponentielles, J. Math. pures et appl., Vol. 67, No. 3, pp. 227-236, 1988.

[15] Unser, P.-G., Thévenaz P. And Aldroubi A., Shift-orthogonal wavelet bases using splines, IEEE Signal Processing Letters, Vol. 3, No. 3, pp. 85-88, 1996.

[16] Smith, Steven W., The Scientist and Engineer's Guide to Digital Signal Process, "http://www.dspguide.com/pdfbook.htm", last access date: 09.21.2006

[17] Strela, V., Denoising via block Wiener filtering in wavelet domain, Birkhäuser Verlag, Barcelona, 2000.

[18] Zhang, H., Nosratinia A., Wells R. O., Image denoising via wavelet-domain spatially adaptive FIR Wiener filtering, in IEEE Proc. Int. Conf. Acoust., Speech, Signal Processing, Istanbul, 2000.

[19] Donoho, D.L., Johnstone, I.M., Ideal spatial adaptation by wavelet shrinkage. Biometrika., 1994

[20] Donoho, D.L., Johnstone, I.M., Adapting to unknown smoothness via wavelet shrinkage, 1995

[21] Zhang, X.P, and Desai, M.T., "Adaptive denoising based on SURE risk" IEEE Signal Processing Letters, vol. 5,no. 10,pp. 265-267, 1998.

[22] Vidakovic, B., Non-linear wavelet shrinkage with Bayes rules and Bayes factors , J. American Statistical Association, 93, pp. 173–179, 1998

[23] Ogden, R.T. "Essential Wavelets for Statistical Applications and Data Analysis", Birkhauser, Boston, 1997

[24] Ogden, R.T., Parzen, E., Data dependent wavelet thresholding in nonparametric regression with change-point applications, pp. 53-70, 1996

[25] Jansen, M., Malfait, M. & Bultheel, A., Generalized cross-validation for wavelet thresholding, Signal Processing, pp. 33-44, 1997

[26] Nason, G.P., Wavelet shrinkage using cross-validation, J.R. Statist. Soc., pp. 463-479, 1996

[27] Stein, C., "Estimation of the Mean of a Multivariate Normal Distribution," The Annals of Statistics, vol. 9, pp. 1135–1151, 1981.

[28] Nist Sematech, "Engineering statistics handbook", "http://www.itl.nist.gov/div898/handbook/eda/section3/eda3666.htm", last access date: 07.12.2006

[29] Jicheng, L., Zhenkang, S., Biao, L., Zhiyong L., Adaptive nonlinear approach for noise removal of impulsive and nonimpulsive noise in images, Proc. SPIE 3073, pp. 435-440, 1997.

[30] Vidakovic, B., Ruggeri, F., BAMS method: theory and simulations, Sankhy, pp. 234–249, 2001

[31] Research Service Branch, "ImageJ**", "**http://rsb.info.nih.gov/ij/", last access date : 08.10.2006

[32] Fiore, L., Corsini, G., Geppetti, L., Application of non-linear filters based on the median filter to experimental and simulated multiunit neural recordings. Journal of Neurosciences Methods, pp. 177-184, 1997

[33] Körner, T. W., Fourier Analysis, Cambridge University Press, Cambridge, 1996.

[34] Calderbank, A. R., Daubechies I., Sweldens W., Yeo B.-L., Wavelet Transforms that Map Integers to Integers, IEEE Press, 1997.

[35] Lee, J. S., Refined filtering of image noise using local statistics, Journal of Computer Graphic and Image Processing, vol. 15, pp. 380-389, 1981.

[36] Kuan, D., Sawchuk, A., Strand, T., and Chavel,, P. Adaptive noise smoothing filter for images with signal-dependent noise, vol. 7, no. 2, Mar. 1985.

[37] Frost, V., Stiles, J., Shanmugan, K., Holtzman, J., A model for radar images and its application to adaptive digital Filtering of multiplicative noise, vol. 4, 1982.

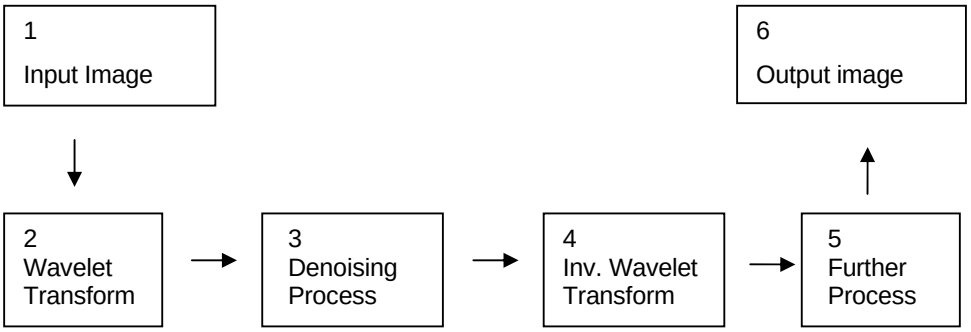
[38] Edmund, Hui-On Ng, "Speckle Noise Reduction via Homomorphic Elliptical Threshold Rotations in the Complex Wavelet Domain", 2005.

[39] Gagnon, L., Wavelet filtering of speckle noise - Some numerical results, 1999

[40] Mansourpour, M., Rajabi, M.A., Blais, J.A.R., Effects and performance of speckle noise reduction filters on active radar and sar images, 2006

APPENDIX A

PROCESS FLOW DIAGRAM



1- Image is handled, two-dimensional array created and its pixel values are copied

2- Wavelet transform is applied. Data sets created by smoothing and detail filters are kept.

3- Denoising process are done on data array contains wavelet coefficients (smoothing data).

4- Inverse wavelet transform is done by using processed wavelet coefficients and detail values. Processed and transformed values are copied to the new two-dimensional array.

5- Further processing is done on new image data file which was reconstructed.

6- Noise taking out from original image, wavelet coefficients and denoised image are displayed.

APPENDIX B

IMPLEMENTED WAVELET MODELS

Haar filter

Daubechies filters (4, 6, 8, 10, 12, 20)

Coiflet filters (2, 4, 6)

Spline filters (2-2, 2-4, 3-3, 3-7)

Pseudocoiflet filter-4

Battle Lemarie filter

Burt Adelson filter

Cohen-Daubechies-Feauveau CDF (N=2,Ntilde=4)