Wavelet Analysis of 3D Medical Imaging Data

S. Koby Taswell

Undergraduate Thesis in Mathematics and Scientific Computation

University of California, Davis

Faculty Advisor: Naoki Saito

January 18, 2023

Abstract

Medical imaging is an important tool for effective medical diagnosis and treatment but is very reliant on signal processing. In recent years, volume resolution and extent of real world space has vastly improved in position emission tomography (PET) imaging with recent developments with whole body PET scanning. As a result, methods for compression, denoising, and more need to be re-evaluated for accuracy and efficiency on larger and increasingly complex datasets. Wavelet analysis is one such tool with the discrete wavelet transform enabling the data to be processed in the transform domain, rather than directly as raw data. In this thesis, we evaluate the use of wavelets in large 3D medical imaging data sets and discuss future uses and considerations for wavelets in medical imaging.

Acknowledgments

I would like to thank my parents for inspiring me, Professor Saito for guiding me, and my friends for cheering me on.

Contents

1 Introduction

In the world of medicine, non-invasive or minimally invasive methods for investigating the body have become a gold standard for diagnosis of disease and injury. Among these methods, medical imaging such as computed tomography (CT), magnetic resonance imaging (MRI), and positron emission tomography (PET) have been used to understand both function and anatomical structure of the body in living patients where invasive biopsy is harmful, far from ideal, or simply unnecessary [1, Ch. 1]. One commonality to all medical imaging methods is the reliance on reconstruction of an image based on the scattering and detection of some form of energy such as radiation, magnetic fields, and more. As a consequence, appropriate processing to remove undesirable error and noise is necessary to find an accurate set of images representing the inner workings of the human body. After successful denoising, other questions arise such as how to find specific features within the volume, how to safely compress the data without losing significant features, and more. One possible solution to these questions is wavelet analysis.

Since their inception in the late 1980s [2, 3], wavelets have been used for a wide variety of applications due to a necessity for more efficient and exact processing methods. Wavelet analysis functions well as a highly adaptable method which can be tailored to a particular signal type and captures both fine and coarse information to understand both little and big picture. Outside of their adaptability, wavelets also well capture a large amount of the original signal within a smaller set of a few values, which can be used to reconstruct the original dataset.

One of the most common uses of wavelets today is image compression as seen in the widely used JPEG 2000 image format [4], a file format used by many modern cameras, computers, and phones. Other uses include structural analysis searching for internal defects in structures [5, 6, 7], power line extraction [8], protein structural analysis [9], solving partial differential equations [10], posture analysis [11], and medical image processing [12, 13, 14]. Each of these different application areas arise from a general need for analysis of complex data, a problem that can be solved via signal processing.

1.1 Multidimensional Signal and Image Processing

In general, signal processing refers to the analysis, modification, and synthesis of signals which contain a set of data as a function of some other variable such as time [15, Ch. 0]. Common examples of signals that we are accustomed to include audio files and pictures, which we will discuss as examples in this section. Audio files convey information about sound as a function of time, a 1 dimensional variable. Pictures convey information about color as a function of 2 dimensional space with both an X and Y axis. We can find a more complex version of both of these signals when looking at movies and films. Movies add the variable of time to a picture, conveying information about color as a function of both space and time, then matches it with an additional output stream of audio intended to be synchronized in time with the displayed image. As humans, we take much of the complexity of these basic signals for granted as our brains are well adapted to handling many streams of information simultaneously. At the same time, we have difficulty picking up on certain pieces of information directly, giving us a need for signal processing algorithms to highlight the desired information.

Prior to discussing specific methods or algorithms, we need to consider some properties of the original data. In specific, these properties are the possible range of values, sample resolution, and error. Range refers to the range of different possible outputs for a given signal. For example, if a picture uses 8-bit gray scale representation in an image, a specific pixel can take on 256 shades of gray, but a color image is typically more complex with perhaps 8 bits to represent each of red, blue, and green creating $(2^8)^3 = 16,777,216$ different possible color combinations. Next, resolution refers to the number of samples within a signal given a certain unit of measurement. Digitally stored music is often recorded at 48kHz or 48000 samples per second while a picture might have a certain number of pixels in each dimension, such as 600×480 for a total of 288000 pixels in what is now considered a low resolution image. Finally, we consider error, best described as the imperfections or inaccuracies introduced by attempting to measure a specific phenomena. Error in an image is typically described as having a grainy appearance when described as noise, but can also come about as an error if taking a picture while moving, incorrectly capturing the desired target.

With a good understanding of what to expect out of our signal, we can consider different ways to analyze, modify, and synthesize a signal. To achieve this, we might filter out certain pieces of signal that meet certain criteria. In an image we might want to find all pixels in a certain color range, removing any values outside of this range. Alternatively, we might try something more complex, searching for edges by looking for points of discontinuity in brightness and color usi ng the Sobel operator [16]. The more complex our method, the more clever an algorithm must be to efficiently compute.

1.2 3D Medical Data

Most medical imaging data takes a number of different forms, but usually we are concerned with an intensity value at a certain position in space as measured by the imaging modality. To describe the data of interest, we have a 3D matrix with each axis corresponding to spatial coordinates X, Y, and Z specified by the scanner's resolution. Each position within the matrix, referred to as a voxel (volumetric pixel), is an intensity value representing different information depending on the imaging modality. For PET imaging we see intensity as a measure of the radiation emitted by a radiotracer, targeting specific molecules of interest for tracking. Thus, areas of high intensity imply a large amount of use or activity in the targeted molecules of interest. In each case of CT, MRI, and PET, we have a major concern of error introduced by the scanning method. For a given scan, there may be issues due to subject movement during the scan or noise caused by radiation passing through and scattering off materials. Due to this, denoising and removal of error is crucial. Once past the issue of noise, we have a few different other areas of interest: segmentation and registration. Segmentation is a form of feature extraction which attempts identify regions of interest such as disease features or specific organ areas. On the other hand, registration refers to the alignment and fusion of two or more different imaging modalities, such as PET and MRI. This enables a physician or researcher to learn more about the patient from the perspective of both imaging modalities in the context of each other. In the case of PET/MRI fusion, we gain understanding of the body's function via PET depending on the radiotracer used while MRI allows for clearer understanding of the anatomical structure within the body.

The need for these methods has been well known for the past decade with various review

articles discussing the current state of medical image processing [12, 13, 14], however, denoising, segmentation, and registration all need to be re-evaluated for the new possibilities in imaging. Some major improvements in these imaging modalities have seen the improvement of resolution with less real world distance represented by each voxel and larger area of the body able to be scanned. In particular, PET has seen significant improvement with the creation of the Explorer total body PET scanner by Badawi and Cherry enabling the higher resolution scanning of whole human body from head to toe [17]. This results in a much larger data size than ever before, so efficient and effective methods are needed to compress or denoise these large data sets.

1.3 Aims

In this thesis, we discuss and evaluate a few different wavelet based techniques for manipulating large 3D medical image data volumes, looking into both effectiveness and efficiency for artificially constructed and real world data. First we will briefly discuss and test data compression via approximation, then continue onto two of the main wavelet based methods for denoising a dataset; hard and soft thresholding. For both of these denoising methods, we look into a number of different parameters which can have an effect on resulting outcomes and test denoising methods for various levels of noise.

2 Wavelet Theory

Wavelets are building blocks, which can be used to describe, learn about, and reconstruct a signal or set of data. To be successful as building blocks, we want them to capture a significant amount of information while not being so specific that they can only capture a particular signal well [18]. If we have a number of different sets of building blocks, we might want to consider what is the best choice of materials for a particular job as different signal types may be better represented by different building blocks. This is in some ways analogous to building a car out of metal, but a house out of wood and stucco; different building materials for different jobs. Since we want to evaluate different methods for processing medical image data, we first should be sure to understand our building materials and different options. In this section, we will discuss some of the basics of wavelet theory. For a more in depth coverage of the topics, see the following books by Wickerhauser, Daubechies, Meyer, and Mallat [18, 19, 20, 21].

2.1 Origins in Fourier Analysis

To understand wavelet analysis, we must look at its origins and where it came from. Various mathematicians contributed to related work in the late 1700s, but Fourier analysis first began with Joseph Fourier in 1822 as basic theory of how to represent an arbitrary function by a series of sines and cosines [22, Ch. 3, Sec. 2]. Utilizing Euler's identity proved to be the correct method, so as explained by Stein [23, Sec. 1.2], we define the Fourier integral or Fourier transform for a function $f(x)$ as:

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i x \xi} dx \tag{2.1}$$

With the inverse transform for $g(\xi)$:

$$g(x) = \int_{-\infty}^{\infty} g(\xi) e^{2\pi i x \xi} d\xi \tag{2.2}$$

This enables us to find the desired series of sines and cosines. If $f(t)$ is a one-periodic function, then its Fourier Series is the infinite sum:

$$f(x) \approx \sum_{k=-\infty}^{\infty} c(k) e^{2\pi i k x}. \tag{2.3}$$

Values $\{c(k) : k \in \mathbb{Z}\}$ are termed the Fourier coefficients of f computed by:

$$c(k) = \hat{f}(k) \stackrel{\text{def}}{=\!=} \int_0^1 f(x) e^{-2\pi i k x} dx \tag{2.4}$$

By transforming our function or signal into a series of trigonometric functions, we can see the different amplitudes of each frequency contributing to our function or signal. We can also perform a simple reduction of data size or compression by choosing a certain set of amplitude coefficients and only considering these coefficients, then when necessary we can utilize the inverse transform to return to the original function or signal domain, and find a close approximation of our original data. One nice property of the Fourier transform is the preservation of energy when transformed. By Plancherel Theorem [24], if f is in $L^2(\mathbb{R})$, then the corresponding $\hat{f}$ in the transform domain is in $L^2(\mathbb{R})$ and $\|f\| = \|\hat{f}\|$. This is relevant to testing and evaluation as it means that modification and processing of the data in the transform domain results in a matched change in energy in the original signal domain. Therefore if we want to evaluate the amount of change in the signal, we can compare the energy of the original data to the processed data in the transform domain, without having to return back to the original signal domain.

Although the Fourier Transform is a very powerful tool, it has two main drawbacks. First, it loses any information about spatial or temporal localization of frequency, depending on the original signal. Secondly, the Fourier Transform uses the specific building blocks of sine and cosine, however, trigonometric functions may not be the optimal choice for all data sets. We can answer these questions with clever selection of basis that is both localized and sufficiently generalizable to find wavelets.

2.2 Wavelet Analysis

Instead of applying a singular transform to the entire dataset globally, suppose we split a signal into partitions then analyze each partition. This allows us to regain some of the lost spatial resolution, however, we greatly increase the amount of computation required by fully analyzing each partition individually. To solve this, we can construct a more clever basis for decomposition that partitions the dataset by means of applying a convolution. Our basis is defined by a generating function, commonly called the mother wavelet. Let $\psi \in L^2(\mathbb{R})$ with a few properties [21, Sec. 4.3]:

(i) ψ has a zero average, $\int_{-\infty}^{\infty} \psi(t) dt = 0$

(ii) ψ is normalized, $\|\psi\| = 1$

(iii) and ψ is centered at $t = 0$.

One way to think of this graphically is that our mother wavelet must be bounded, oscillatory, and localized such that prior to oscillation our function is at zero, then returns to zero after oscillation. With ψ defined, we can find a family of wavelets by shifting and scaling ψ:

$$\psi_{u,s}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right).$$

In a simple 1D case, u shifts the center of ψ and s shifts the width of ψ. We can then find our transform of a signal f via inner product between our basis and f, or equivalently the convolution of the two.

$$Wf(u,s) = \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)dt \tag{2.5}$$

In the discrete case as is needed for signal processing, we utilize techniques which came about from a combination of the work by Meyer, Mallat, Daubechies, and more. This comes about as a multiresolution analysis combined with a set of filters based on our generating wavelet ψ [18, Sec. 6.1]. A single level of the 1-dimensional discrete wavelet transform (DWT) on a signal x can be described as follows. Given a low pass filter g and a high pass filter h, we convolve our signal x with each of g and h:

$$y_{\text{low}}[n] = \sum_{k=-\infty}^{\infty} x[k]g[2n-k]$$

$$y_{\text{high}}[n] = \sum_{k=-\infty}^{\infty} x[k]h[2n-k]$$

After finding each of $y_{\text{low}}, y_{\text{high}}$, then downsample the resulting signal by a factor of 2. Our downsampled y_{low} values are terms approximation coefficients, sometimes referred to as scaling coefficients, and our downsampled y_{high} values are referred to as detail coefficients. To find the next transform level, we perform the same operation, this time using our previous level's approximation coefficients as our signal x. This can be repeated on a signal of dyadic length until resulting in the maximum level transform of only one remaining approximation coefficient. Similarly, we can perform these operations in reverse using inverse filters to reconstruct the original dataset based on each the coefficients of each transform level.

This series of operations works well for a 1D signal, however, it is insufficient for higher dimensional datasets as it does not preserve the spatial resolution or structure. To handle a 2D or 3D dataset, we need to generalize this method to higher dimensions by applying our filter pair to both y_{low} and y_{high} another time for each dimension, this time reoriented to the axis being analyzed. Consider a 2D case of an image with (x, y) spatial dimensions. First we filter for high and low pass along the X dimension, then we filter again for high and low passes in the Y dimension, for each of our previous results. This results in four resulting coefficient sets based on low pass L and high pass H results combining to find LL, LH, HL, and HH where our LL set remains as our approximation

coefficients, while the remaining three coefficient sets LH, HL, HH are our detail coefficients corresponding to that level. Similarly to 1D, to find a higher level transform, one would perform the operation again on the previous level's approximation coefficients to find the next desired level.

3 Evaluation of Techniques for 3D Volumes

Since wavelet analysis techniques pose challenges for large 3D data volumes, we can evaluate different techniques on a variety of datasets which could reasonably represent the variety of applications we might consider. In this section, we will first briefly introduce the data for evaluation, discuss choice of mother wavelets, and see different techniques applied in action. Software for experimentation was developed in Julia 1.8.1 and ran on a two different machines, first Windows 10 laptop with Intel i7-11370H, 40 GB of RAM, and NVidia RTX 3070 and second Windows 11 mobile workstation with Intel I9-12950HX, 128GB of RAM, and NVidia RTX A5500. The Wavelets.jl package available on GitHub was used for computing the DWT of a signal and selection of wavelet types. The code repository for reproducing any experiment is available upon request.

3.1 Data Creation and Selection for Evaluation

To properly evaluate different techniques, we need to consider some traits we can assess. Since the goal of this thesis is focused on medical imaging where we are finding a 3D structural representation of the body, we will restrict our possible data structures to non-periodic shapes. Some guiding questions for underlying properties to consider include:

- Are there discontinuities within the data?
- Is the data noisy?
- How large is the set of data?

To answer these questions, we will consider both constructed and real world data, then in each case add noise to the dataset.

3.1.1 Constructed Data

For our first constructed data set, we can consider a ball with a normal distribution of overall density, referred to as the 'simple ball'. As we move away from the center of the ball along a radius, we find a decrease in intensity. One real world example of this density distribution would be the distribution of mass within the sun, where the core of the star is most dense while the furthest regions are least dense. Mathematically, we can define this as a multivariate normal distribution with equal variance in each dimension, centered at a particular point in 3D space. The standard Gaussian (or normal) distribution in 1D with terms μ and σ for mean and variance is given by the following:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.1}$$

Since we want a reasonable range of intensity values on a range from $[0, 1]$, we can remove the scaling factor of $\frac{1}{\sqrt{2\pi\sigma^2}}$, then by treating each dimension (x, y, z) independently, we can find a resulting distribution:

$$f(x, y, z) = p(x)p(y)p(z) = (2\pi\sigma)^{(3/2)}p(x)p(y)p(z) = e^{-\frac{1}{2\sigma^2}((x-\mu_1)^2+(y-\mu_2)^2+(z-\mu_3)^2)} \tag{3.2}$$

For our purposes, we will work with a volume of dimensions $128 \times 128 \times 128$ with data centered at the middle of the volume and a resulting rendering can be seen in figure 3.1 (a). The white 3D axes refer to the spatial coordinates of intensity within the volume, this convention will be maintained throughout this thesis. Additionally, note that with this distribution as the generator of our dataset, we have no zeros positions which result in a true 0 value.

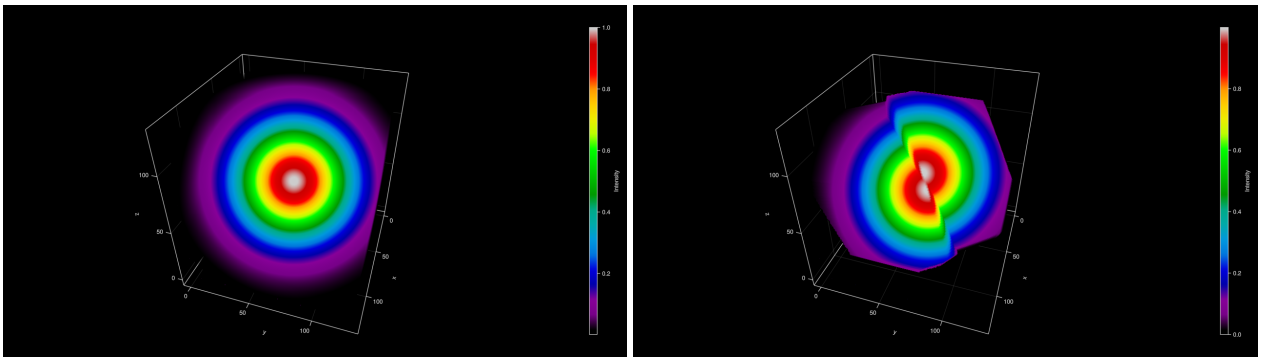
For our second constructed dataset, termed 'split ball' we introduce both discontinuity and zero values, two desirable properties as this is more similar to medical data. We can create a discontinuous region by "cutting" our simple Gaussian ball along a plane, then sliding each hemisphere in an opposite direction. The choice of the slicing plane does have some significance. If we were to slice along a plane containing any axis, our computation in that dimension would be greatly simplified as there would be minimal discontinuity in that dimension, if at all. Similarly, we cannot consider the 45° angle either as that is a special angle which could simplify certain computations when utilizing trigonometric functions. Rather than attempting to slice directly along some plane, we can first slice along a plane, then rotate our data within our volume to shift the plane into a particular orientation. This method results in the introduction of zero values if performing the rotation within the original matrix as data points outside of the matrix boundaries need to be rotated into the volume. These values do not exist, thus the function applied by the Rotations library introduces zero values for non-existent values depending on chosen settings. This is desirable as real data may be processed prior to manipulation to quantize and compress for easy storage, which changes many extraneous values outside of the scanned subject's body to zero. One downside to this method is that when performing DWT on this volume, the irregularities in the paired faces of left-right, bottom-top, and front-back in the edges of the 3D cubic matrix are handled as regions of discontinuity by Wavelets.jl. This is caused by Wavelets.jl performing a periodic convolution, such that when reaching an edge of a volume, the convolution operation utilizes values on the opposing face to find the resulting value of the convolution at that index. Due to this, a significant amount of wavelet coefficient size in the DWT of the split ball can be attributed to Our resulting volume when rendered can be seen in figure 3.1 (b).

3.1.2 Medical Data

Real world examples of medical data were provided by Dr. Simon Cherry of UCD with de-identified data of subjects scanned on the uExplorer PET whole body scanner [17]. Three different data sets were provided each scanned using F-18-FDG radiotracer corresponding to glucose activity within the body. Two data sets at resolution $256 \times 256 \times 828$ were provided for Subjects 1 and 2, then a second data set for Subject 1 was provided at higher resolution of $512 \times 512 \times 1656$. For all medical datasets, each voxel was represented by a 16-bit unsigned integer prior to computation.

Due to limitations of Wavelets.jl library package, data could only be handled by DWT for dimensions with all non-dyadic lengths or all dyadic lengths, not a combinations of both dyadic

Figure 3.1: Rendering of Constructed Data



(a) Simple Ball (b) Split Ball

White 3D axes lines denote spatial coordinates of intensity within the data volume

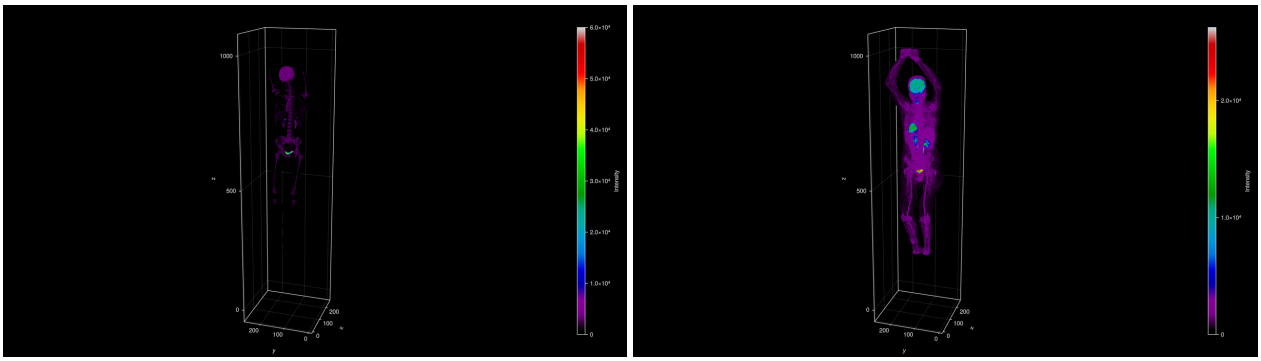
Table 3.1: Signal Value Distribution in Each Dataset for Computation

Dataset	Size	Mean	Std Dev	Min	Max	% Equal to 0
Simple Ball	$128 \times 128 \times 128$	0.1985	0.1970	0.0016	1	0
Split Ball	$128 \times 128 \times 128$	0.1828	0.2087	0	1	29.36
Subject 1 Small	$256 \times 256 \times 1024$*	23.8201	189.1147	0	60000	89.23
Subject 2 Small	$256 \times 256 \times 1024$*	75.4223	403.6065	0	26252	73.68
Subject 1 Large	$512 \times 512 \times 2048$*	23.7376	188.1734	0	59999	88.7

$*$ denotes volumes padded with zeros so as to function within Wavelets.jl limitations

and non-dyadic. To compensate, data was padded prior to computation to the nearest dyadic value of 2^n in each dimension as necessary. There were some interesting features of this data visible in summary table 3.1, notably large percentage of each original volume had zero values prior to padding. On average, zero padding to dyadic dimensions resulted in between 2-5% increase in the percentage of the volume composed by zeros. To briefly describe each dataset, Subject 1 had a much lower average signal around 24, with peak signals concentrated near the lower half of the body. Subject 2 had a higher average signal than Subject 1 at around 75 with greater dispersion of intensity across the body, significantly lower peak values at 26252 down from 60000 and more than double Subject 1's standard deviation. The larger, higher resolution version of Subject 1 had almost identical features as the lower resolution version. Since Subject 1 Large had double the length in each dimension, it was 8 times as large as the already relatively big dataset in both Subject 1 and Subject 2 making computation experiments involving high resolution for Subject 1 extremely costly. As a result, only small resolutions will be evaluated for compression and denoising under the assumption that Subject 1 Small at a lower resolution is sufficiently representative of the same dataset at higher resolution. Renderings of each small medical dataset can be see in figure 3.2.

Figure 3.2: Volume Rendering of Medical Datasets



(a) Subject 1 (b) Subject 2

3.1.3 Including Noise

Since we want to consider noisy volumes as well, we need to introduce noise into our data volumes. This is not necessarily an accurate comparison to a real world situation as noise in PET and other medical imaging may not have the same dispersion, pattern, or level, however, for the sake of standardization and simplicity we will keep the noise pattern identical across all experiments. Since different datasets have different distribution properties, we do have to consider a variety of noise levels. Thus, for a given dataset we should use various values for variance σ depending on the properties of the original signal, without increasing the overall energy of the dataset with an increased mean. Finally we must keep in mind that considering a specific PET scan as a clean noiseless dataset is somewhat of a mistake. Due to the method of data collection, PET data must be assumed to already have noise even when handling a seemingly clean and reconstructed dataset such as those provided by utilized in this thesis. Thus introducing noise to such a dataset is even less ideal, however, we proceed making the assumption that the medical data provided is as close enough to 'truth' as is reasonable.

Using the Julia library Noise.jl [25], we can introduce additive Gaussian noise using the `add_gauss(X,var,avg)` function provided by Noise.jl library with variance σ and average μ which default to $\sigma = 0.1$ and $\mu = 0$. Following the principle of scaling noise variance based on the data distribution, we will use the variances $\sigma = 0.1, 0.2, 0.3$ for constructed data and $\sigma = 0.1, 10, 50, 100$ for medical data. Although each noise level is visible in constructed data, variance values needed to be much higher to be perceptible to the human eye when inspecting the entire volume, thus larger values of $10, 50, 100$ were chosen while still computing a baseline minimal amount of noise at $\sigma = 0.1$ only noticeable by computation. In both cases of medical and constructed data, inclusion of noise results in a volume with no zero values which will be used for comparison against previously described datasets.

3.2 Wavelets for Evaluation

Although there are a large number of possible choices of wavelet for evaluation of medical data, in this thesis we will evaluate a selection of those available in the Wavelets.jl library. Four of the eight wavelets available in Wavelets.jl were chosen; Haar, Daubechies, Coiflet, and Symlet. Following the classification method utilized by Wavelets.jl, the nomenclature for a given wavelet is basis name, then number, where the value refers to the number of vanishing moments. Haar was selected as one of the simplest wavelets for computation, with only one vanishing moment. Daubechies, Coiflet, and Symlet were chosen as widely known and studied wavelet types well represented at a variety of different vanishing moment counts and filter lengths. Wavelets excluding Haar were chosen at 4 vanishing moments as the lowest possible count of vanishing moments common to all three wavelet types of Daubechies, Coiflet, and Symlet within the Wavelets.jl library. Transformations were performed on each dataset for each wavelet type to evaluate both timing tests and to see how well the dataset was captured in the largest coefficients.

In general, Coiflet 4 was the slowest transform across all datasets while Haar was the fastest across all datasets. These timings are largely expected as Haar uses the simplest filter to compute the discrete wavelet transform with only two filter coefficients, while Coiflet has the longest filter length at three times the number of vanishing moments using 12 filter coefficients. Both Daubechies and Symlet wavelets have similar properties with respect to filter length with filter length equal to two times the number of vanishing moments, resulting in largely similar computation times. Interestingly, Symlet 4 was slightly slower than Daubechies 4 for datasets with larger amounts of discontinuity, so large medical dataset, Daubechies 4 was 2.2 seconds faster than Symlet 4. These timings can be seen in Table 3.2 which represent evaluation on a the lower power laptop, however, the general trend remained the same when evaluating on the higher power mobile workstation albeit faster given the stronger processor available on this machine. Lastly, these transform timings represent a maximum level transform as computed by the maximum levels condition by the Wavelets.jl library. This did result in the final transform level being computed on set of approximation coefficients of smaller dimensions than the filter lengths of Daubechies 4, Symlet 4, and Coiflet 4, which may be undesired. The compression and denoising experiments discussed in this thesis also used a maximum level transform.

In all datasets, very few resulting wavelet coefficients were larger than 1, while the number of wavelet coefficients in each dataset within error precision or at 0 was associated with sparsity of the original dataset. As described above, the simple ball had no zero values and its resulting wavelet coefficients were largely not within error. For the remaining datasets, the flat line in each curve represents coefficients at zero, corrected to error precision for appropriate plotting on log scale. In the split ball and both medical datasets, Haar resulted in the largest number of zeros but also had slightly larger coefficients on average. Coiflet 4 resulted in the largest number of wavelet coefficients in all sparse datasets while both Daubechies 4 and Symlet 4 performed similarly. These results can be seen in 3.3.

Table 3.2: Wavelet Transform Timing

Dataset	Transform Timings			
	Haar	Daubechies 4	Coiflet 4	Symlet 4
Simple Ball	47.0ms	106ms	134ms	105ms
Split Ball	41.7ms	95.9ms	123ms	96.2ms
Noisy Simple Ball	49.6ms	103ms	134ms	102ms
Noisy Split Ball	50.5ms	104ms	129ms	105ms
Subject 1 Small	2.38s	3.99s	4.89s	4.11s
Subject 2 Small	2.31s	4.10s	4.96s	4.12s
Subject 1 Large	36.2s	49.7s	56.3s	51.9s

3.3 Error and Noise Metrics

To assess accuracy and denoising, we use relative norm error and peak signal to noise ratio (PSNR). We define relative norm error by the standard L^2 norm. Given $\boldsymbol{f}$ original signal and $\boldsymbol{g}$ as the modified signal, both in the original or transform domain, we have

$$err = \frac{\|\boldsymbol{f} - \boldsymbol{g}\|_2}{\|\boldsymbol{f}\|_2}.$$

This allows us to see how similar the two volumes are in terms of total energy. For compression specifically, ideally the error is as low as possible for a certain approximation with a number of coefficients. PSNR can be defined via mean squared error, however, following Irion's use of signal to noise ratio [26], we will use PSNR as defined the following. Given a noisy signal $\boldsymbol{g} = \boldsymbol{f} + \boldsymbol{\epsilon}$ where ϵ is error or noise and $\boldsymbol{f}$ is the known original true signal.

$$PSNR = 10 \log_{10} \left(\frac{\max(\boldsymbol{f})^2}{\|\boldsymbol{g} - \boldsymbol{f}\|_2^2} \right) \tag{3.3}$$

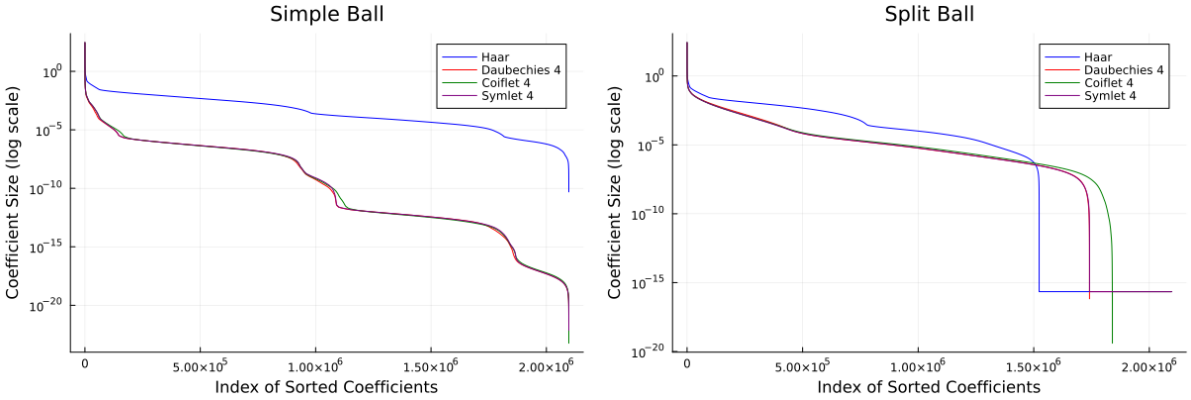
Here $\|\boldsymbol{g} - \boldsymbol{f}\|_2^2$ is representative of the energy of the noise as removal of the true signal from the noisy signal returns only the noise added to the volume. These metrics are representative of the mathematical value representation of each array, so to verify that a certain marked decrease in relative error or increase in PSNR should be evaluated visually to ensure that human vision can observe improvements. There are a number of different metrics used to consider how well an image would appear to a human such as Structural Similarity and more [27, 28], however, these metrics were not utilized in this project.

3.4 Compression

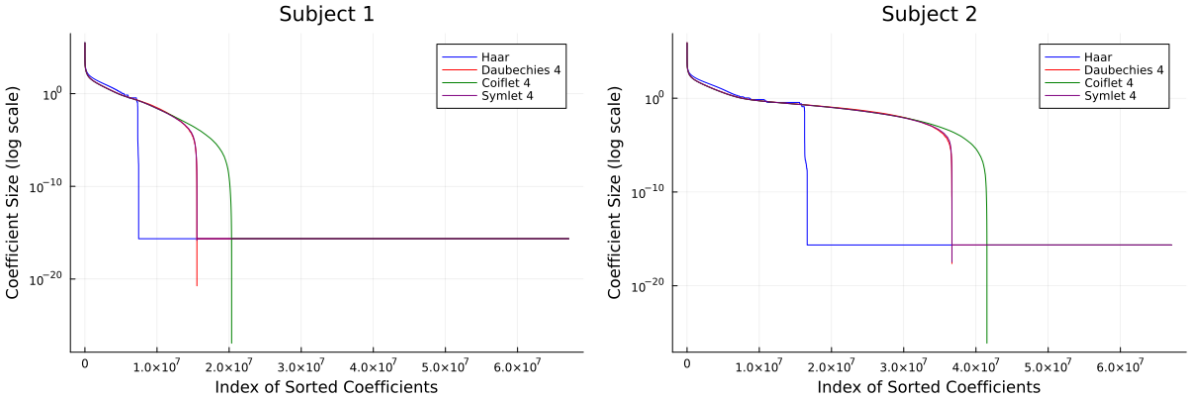
Wavelets can be a useful method for data compression as the majority of the original signal is well captured within a much smaller number of wavelet coefficients. Since compression is not typically a huge question in medical data, we will briefly discuss the simplest of transform based compression

Figure 3.3: Size of Wavelet Coefficients

(a) Constructed Datasets at $128 \times 128 \times 128$ Resolution



(b) Medical Datasets at $256 \times 256 \times 828$ padded to $256 \times 256 \times 1024$ Resolution

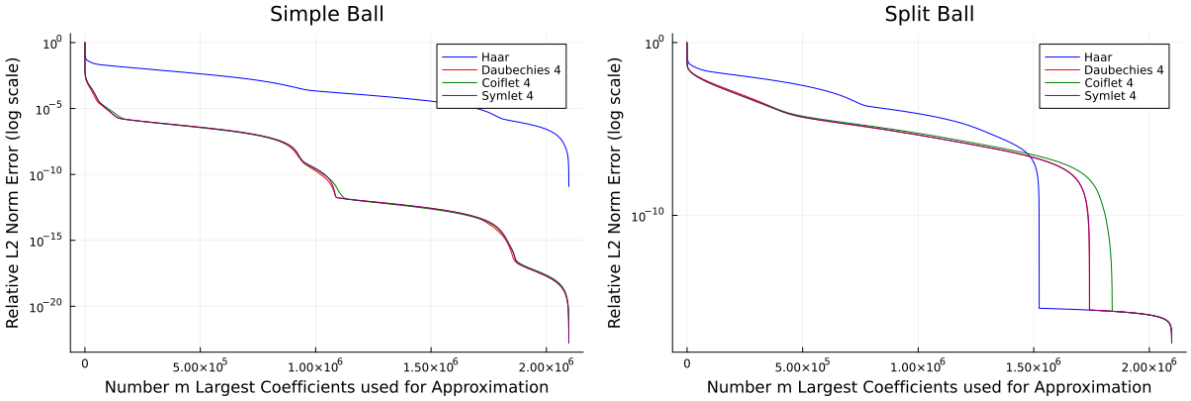


methods, approximation via a number of wavelet coefficients. Suppose we perform a maximum level wavelet transform on a certain dataset, resulting in N coefficients in total. As above in our initial transform tests, we can sort these coefficients by absolute size. Then suppose we attempt a reconstruction of our original dataset with 1% largest coefficients, so we set coefficients outside of this range to zero and proceed with m coefficients. In figure 3.4, we can see approximation of the original signals via each wavelet type for coefficient counts $m \in [1, N]$ evaluated for relative error. In for constructed datasets, we see that Haar performed the worst with the highest error across all datasets at every choice of coefficient count. While for medical datasets, Haar only performed the worst for the majority of its non-zero coefficients, once past the threshold of 0 coefficients as can be seen in figure 3.3, the corresponding dataset reached error precision. The remaining 4 vanishing moment wavelet types had roughly similar performance for most regions of constructed data, but followed a similar pattern of rapidly approaching zero at a point depending on number of zero coefficients. Coiflet 4 reached the lowest error point when evaluating for approximation

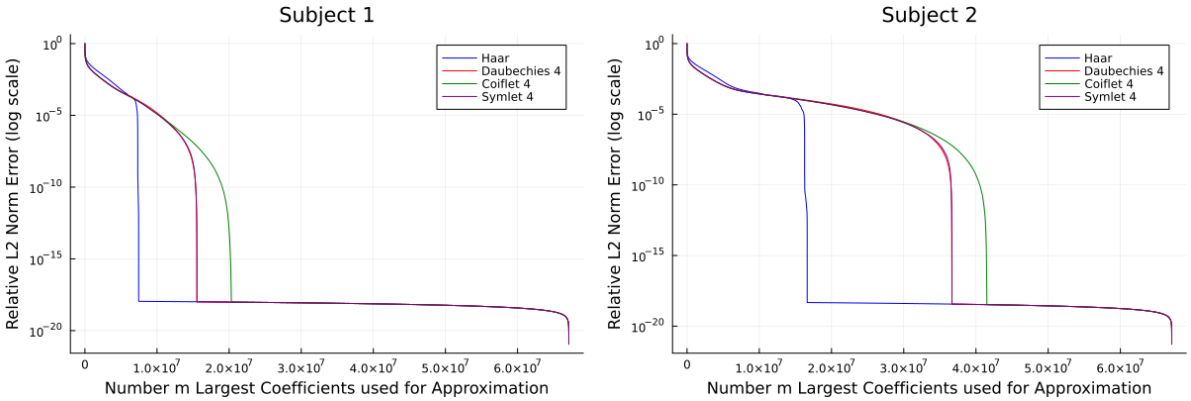
via all non-zero coefficients in datasets with a number of zero coefficients. Specific to compression, being able to encode the maximal amount of information in the fewest amount of coefficients is preferred, thus although has a larger amount of error at most largest coefficient counts m, Haar has the fewest coefficients equal to zero after initial transformation thus could be a reasonable option. When these results are considered in combination with our timing tests in table 3.2, we see the trade off of maximum accuracy versus speed efficiency.

Figure 3.4: Approximation via m Largest Coefficients

(a) Constructed Datasets



(b) Small Medical Datasets



3.5 Denoising

There are many different methods for wavelet based denoising of a signal, but in general we can simplify to two categories; hard or soft thresholding. In general, both methods leverage the property of wavelets that majority of features of a signal are well captured by a fewer large coefficients, while very fine detail features such as those caused by noise would likely be captured in small coefficients. Hard thresholding is a simple thresholding method which only retains values larger than a specific

chosen threshold T, setting coefficients smaller than T to zero. These general methods can be applied to any number of different transform methods, including Fourier coefficients, however, we have utilized them for wavelet transform denoising following in the footsteps of past authors [29,30]. In many ways, this is identical to approximation with fewer coefficients, however, in a noisy dataset we remove these small coefficients which mostly capture noise in an attempt to capture the 'true' signal. There are some short comings with this method, notably that a large number of small coefficients may represent both fine details in the true signal and noise. We can attempt to remove just the contribution by noise while retaining the contribution from true signal by performing soft thresholding instead. Soft thresholding is a more careful operation which removes coefficients less than T, but also shrinks other coefficients towards zero by the magnitude of the threshold. There are different schemes for performing soft thresholding, but we will be using the method defined by Irion and Saito [26] which performs soft thresholding based on approximation versus detail coefficients. For wavelet coefficients c_l^j with $j \in [0, j_{\max}]$ corresponding to transform level and $l \in [0, l_{\max}]$ denoting index within the transform level, we soft threshold coefficients by

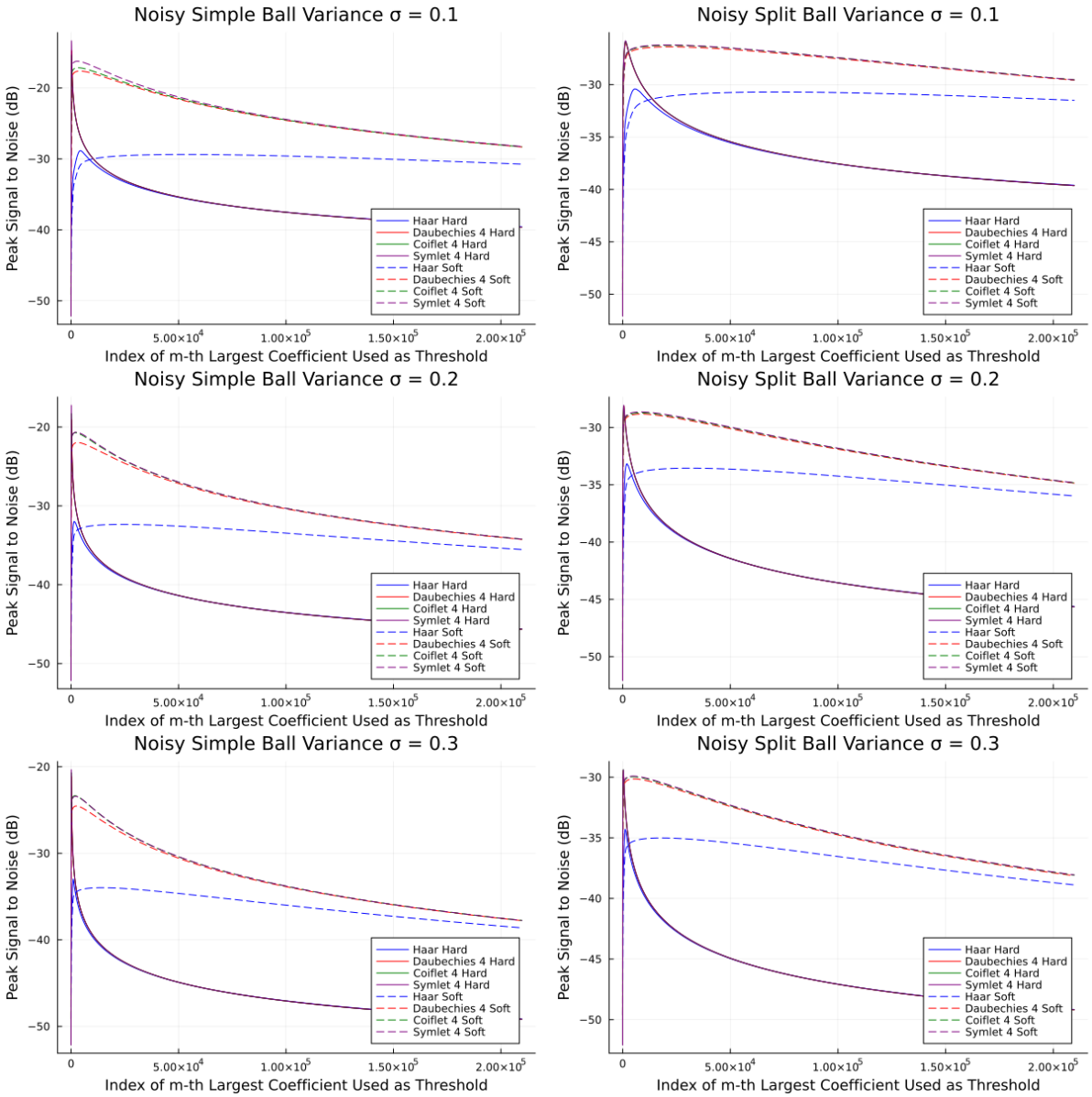
$$\tilde{c}_l^j = \begin{cases} c_l^j & \text{if } l = 0 \\ \text{sign}(c_l^j)(|c_l^j| - T)_+ & \text{otherwise} \end{cases}.$$

In plain language, we retain the coefficients corresponding to the approximation coefficients at every transform level as is, then shrink all other coefficients towards zero by T to a minimum of zero. By retaining the approximation coefficients, we maintain and preserve data averages corresponding to the entire transformed dataset, which should be minimally effected by noise. This improves on conventional soft thresholding which globally shrinks all values towards zero, however, could possibly be further improved by careful selection of transform level and/or level specific thresholding such that a threshold t is selected for each level based on that level's coefficients.

For either case of soft or hard thresholding, selection of appropriate threshold T is crucial with many different methods for choice of an optimal value for T. If T is too large, we lose elements of the original signal, while if T is too small, we eliminate too little of the noise, and each type or method of thresholding may have certain nuances on a specific dataset. For simplicity, we choose our threshold T from the wavelet coefficients themselves, thresholding based on m-th largest coefficient similar to our approximation for compression.

Figures 3.5 and 3.6 display the results comparing hard and soft thresholding across each of the four selected wavelet types for different noise variances in a given dataset. For hard thresholding, across all datasets and levels of noise variance, we have a very large spike in increased PSNR for large thresholds equal to some of the largest coefficient within the transformed data. This spike is characterized by a much more sharp increase in constructed data and a more smooth peak in medical data. Soft thresholding on the other hand, had a much more smooth curve across all datasets, never reaching the sudden peak as is apparent in hard thresholding. In general, soft thresholding outperformed hard thresholding for almost all thresholds tests, however, within the limited range of high spike values at large thresholds, hard thresholding outperformed soft thresholding on both PSNR and relative error metrics. When inspected in visual renderings, hard thresholding was vaguely recognizable while soft thresholding resulted in a clear improvement with maintained clear distinction between small features such as rib bones for Subject 1 and other regions for Subject 2 as

Figure 3.5: PSNR in Denoising when T is m-th Largest Coefficient in Constructed Datasets



seen in figure 3.7. So although hard thresholding is mathematically similar to the original volume by relative L^2 norm error with low measurements of noise for a small set of thresholds, it is a poor choice of technique due to too much distortion of the image.

Next considering the different wavelet types, we see that although Haar is our worst wavelet choice for the constructed data in general, Haar reaches a point of greater accuracy in comparison

to other wavelet choices on a small range of thresholds for hard thresholding in the low variance case. Despite this, Haar does not reach the same maximum amount of noise removed as other wavelet choices. Specific to Haar in soft thresholding, Haar was the least sensitive to threshold selection, with the smallest amount of variance in results across a wide variety of thresholds. This implies that if using soft thresholding on Haar wavelet coefficients, exact threshold selection once lower than a certain value, will result in a respectable amount of noise removal, especially when compared to hard thresholding with Haar which is very sensitive to threshold selection. For both hard and soft thresholding, Daubechies 4, Coiflet 4, and Symlet 4 outperformed Haar for almost all tested thresholds across all variances and datasets. Out of these three wavelet types, performance is almost identical, with Symlet 4 usually out performing Daubechies 4 and Coiflet 4 by a very small margin (≈ -0.000001 for relative error, $\approx +0.02$ for PSNR), with the difference between these three wavelet types changing based on variance and dataset. Comparing datasets across by variance level in noise, the difference between the extrema in soft and hard thresholding decreased as variance increased. Each of these methods showed worsening results based on variance with greater amounts of reintroduction of noise for medium to small threshold choices. Additionally, at medium to small threshold choices, the difference in soft thresholding accuracy across different wavelet types including Haar was greatly reduced. These results are mirrored in approximation error which can be seen in supplemental figures S1 and S2.

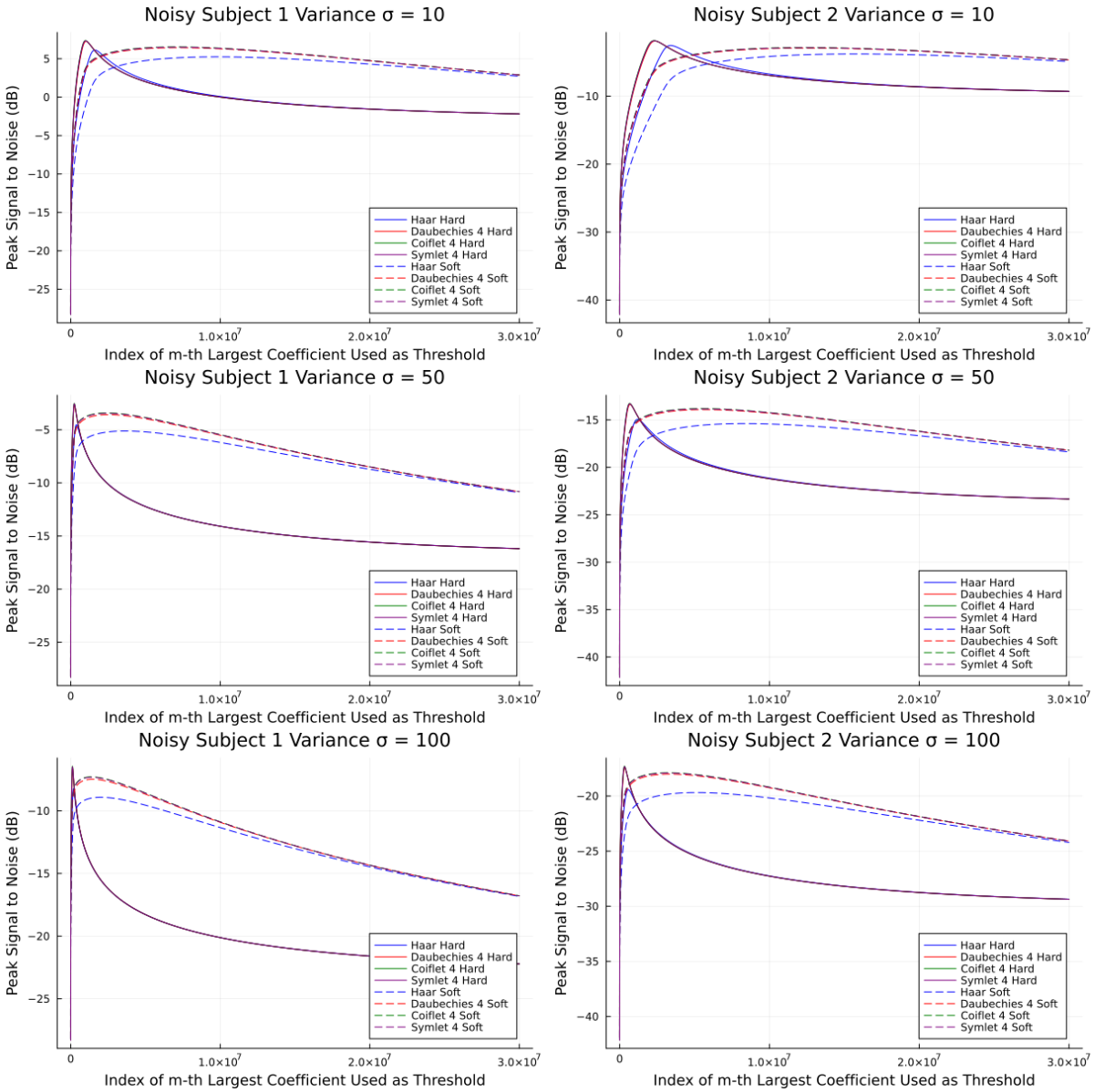
When looking at the visual rendering of soft thresholding and noise, certain bodily features unnecessary to medical evaluation of organs are more noticeable with the introduction of noise most notably the area lower body around the hips of the subject. Although these features are not fully removed by soft thresholding, denoising does reduce how visible these regions are.

3.6 Discussion

For every computational method, there is the question of speed versus accuracy. In this section, we will touch upon this question. As a brief recap, our least complex wavelet type of Haar was fastest to compute in comparison to all other choices, with Coiflet 4 as the slowest transform to compute. According to accuracy metrics of relative error, Symlet 4 had the best performance in most compression cases, beaten by Haar for a some tests in compression. For denoising, soft thresholding outperformed hard thresholding in almost all threshold choices with Symlet 4 seemingly being the best wavelet choice for denoising. In all denoising methods, the performance decreased as variance level in noise increased. Yet, most improvements made by more complex wavelet types were relatively small in comparison to Haar, so can we justify using a method which requires a 1.5 to 2.5 times slower transform compute time depending on size and wavelet type. In what cases, can Haar be sufficient enough levels of accuracy such that it suffices and a more complex method is undesirable. Certainly, for some cases of compression, Haar appears to be a very good option, but the answer is not as clear for denoising.

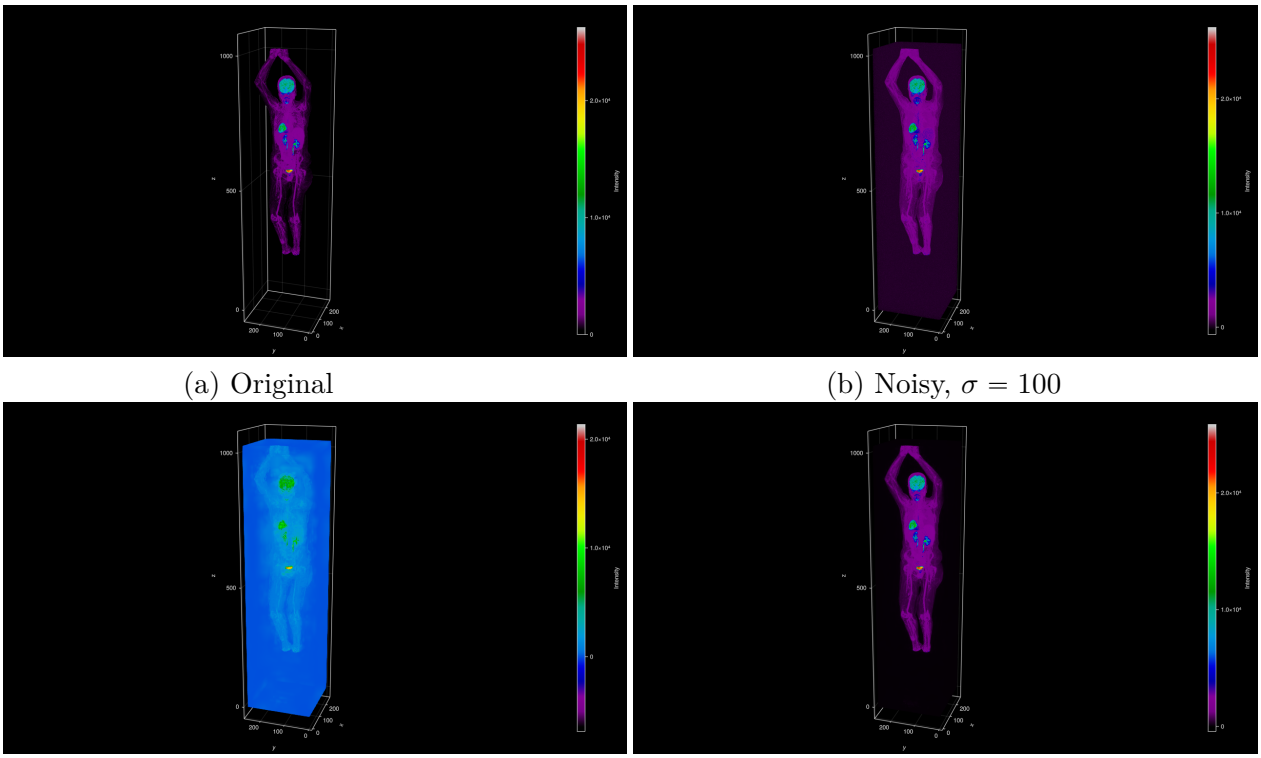
Practical use of a wavelet based denoising method requires the use of an estimator to determine what threshold value to apply via soft or hard thresholding. A number of different options are available such as the double-elbow method by Irion and Saito [26], universal threshold value discussed by [29], and more. If these threshold estimators or a new estimator falls within the peaks for soft thresholding for a given wavelet choice, then the practical selection of a threshold may

Figure 3.6: PSNR in Denoising when T is m-th Largest Coefficient in Medical Datasets



make a certain method and wavelet transform pair more desirable. As briefly touched on when discussing specific thresholding schemes, the method tested in this experiment represented global hard threshold and soft thresholding of all detail coefficients. Other thresholding schemes such as Donoho's SureShrink algorithm, which use a threshold specifically chosen for each transform level may also be superior to the methods tested in this thesis. Finally, to fully understand these

Figure 3.7: Rendering of Denoising via Thresholding in Subject 2



(a) Original

(b) Noisy, $\sigma = 100$

(c) Hard Threshold with Symlet 4

(d) Soft Threshold with Symlet 4

findings, we need to consider a wider range of both parameters and parameter selections. Once we have fully considered enough parameters and computation speed is desirable, then Haar may be a good choice for very large medical datasets. However, for many evaluations the medical world the maximum level of accuracy is preferred. Considering this point, how can we adjust existing methods to enable the practical use of sophisticated denoising schemes paired with more complex wavelet filters?

In this thesis, we assume Wavelets.jl to employ the most optimal methods the Julia programming language has to offer. If we change this assumption slightly to only assume the computation algorithm is well optimized, we can consider the possibility of accelerating 3D wavelet transforms by parallelization of tasks. In recent years as graphics processing units (GPUs) have improved to become more and more powerful, GPU parallel processing has garnered increased interest as a method for greatly accelerating computations by parallelization, separating individual computations across thousands of GPU cores to be simultaneously processed. With GPU programming supported by Julia via the AMDGPU.jl and CUDA.jl [31] libraries, the creation of parallel processing enabled functions for the discrete wavelet transform and subsequent wavelet based denoising may make the speed difference between these different wavelet types negligible, making more complex wavelet filters preferred for wavelet based denoising techniques. Some existing solutions have come about

to accelerate DWT transforms such as Quan's hybrid parallelism based method via CUDA [32] and Chen's Slice method for the lifting based variant of DWT [33], but as of yet no method is available in Julia. If the concern of computation time can be ameliorated, small changes in accuracy and denoising become more viable for practical use.

4 Conclusion

In this thesis, we have investigated an initial evaluation of wavelet based techniques for compression and denoising in large 3-D medical datasets such as whole body PET imaging. Our investigation considered two parameters for compression, data structure and wavelet type, and four parameters for denoising: thresholding method, wavelet type, data structure, and noise level. More complex methods such as soft thresholding paired with more complex wavelet filters such as Symlet 4 result in both higher accuracy measurements and improved readability upon reconstruction, however, come at the cost of computation time in very large datasets.

For most real world cases, a slightly longer compute time is not a problem. High accuracy in measurements and tests is crucial for assessment of patients, but at what point is a less complex wavelet transform sufficient, satisfactory and perhaps preferred? If paired with more sophisticated methods than used in this initial series of experiments, a simpler wavelet transform such as Haar may prove to be acceptable. Alternatively, we may find that as is indicated in this investigation, more complex wavelet would be preferred due to higher accuracy. To fully answer this question in medical image denoising, a more thorough series of tests should be performed to with a wider selection of already chosen parameters as well as additional conditions to be considered. This would include further variation in datasets, noise pattern, wavelet type, thresholding scheme, and threshold selection. Secondly, other metrics which evaluate a volume based on structure or human vision would enable more careful evaluation of the resulting outcome, regardless of exact level of noise removed. Once more aspects have been evaluated for large medical datasets, a more conclusive answer to this question can be determined.

Denoising experiments are only one method within an overarching toolbox for image processing techniques on medical volumes. Various software applications enable segmentation, feature extraction, image registration and more which can all be performed via wavelet based methods. The question of optimal versus satisfactory wavelet and technique for each of these algorithms for medical data can be considered. Finally, if we want to support these techniques within Julia, a more comprehensive and robust code library should be built to better enable researchers to further explore these questions both in medical data specifically and in other datasets.

5 References

[1] M. A. Haidekker, *Medical Imaging Technology*. Springer New York, 2013.

[2] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Communications on Pure and Applied Mathematics*, vol. 41, pp. 909–996, Oct. 1988.

[3] S. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 674–693, July 1989.

[4] D. Taubman and M. Marcellin, "JPEG2000: standard for interactive imaging," *Proceedings of the IEEE*, vol. 90, pp. 1336–1357, Aug. 2002.

[5] A. Katunin, M. Dańczak, and P. Kostka, "Automated identification and classification of internal defects in composite structures using computed tomography and 3D wavelet analysis," *Archives of Civil and Mechanical Engineering*, vol. 15, pp. 436–448, Feb. 2015.

[6] A. Wronkowicz-Katunin, A. Katunin, M. Nagode, and J. Klemenc, "Classification of cracks in composite structures subjected to low-velocity impact using distribution-based segmentation and wavelet analysis of x-ray tomograms," *Sensors*, vol. 21, p. 8342, Dec. 2021.

[7] G. Kilic, "Wavelet analysis and NDT for condition assessment of historic masonry bridge," *Structures*, vol. 45, pp. 275–283, Nov. 2022.

[8] A. Fryskowska, "Improvement of 3D power line extraction from multiple low-cost UAV imagery using wavelet analysis," *Sensors*, vol. 19, p. 700, Feb. 2019.

[9] N. C. Benson and V. Daggett, "Wavelet analysis of protein motion," *International Journal of Wavelets, Multiresolution and Information Processing*, vol. 10, p. 1250040, July 2012.

[10] O. V. Vasilyev, D. A. Yuen, S. Paolucci, R. Gruber, and J. Rappaz, "Solving PDEs using wavelets," *Computers in Physics*, vol. 11, p. 429, Oct. 1997.

[11] L. Deng and Y. Pu, "Analysis of college martial arts teaching posture based on 3D image reconstruction and wavelet transform," *Displays*, vol. 69, p. 102044, Sept. 2021.

[12] D.-Z. Tian and M.-H. Ha, "Applications of wavelet transform in medical image processing," in *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826)*, IEEE, 2004.

[13] Y. Jin, E. Angelini, and A. Laine, "Wavelets in medical image processing: Denoising, segmentation, and registration," in *Handbook of Biomedical Image Analysis*, pp. 305–358, Springer US, 2005.

[14] A. Pizurica, A. Wink, E. Vansteenkiste, W. Philips, and B. Roerdink, "A review of wavelet denoising in MRI and ultrasound brain imaging," *Current Medical Imaging Reviews*, vol. 2, pp. 247–260, Feb. 2006.

[15] R. Priemer, *Introductory Signal Processing*. Singapore: World Scientific, 1991.

[16] P.-E. Danielsson and O. Seger, "Generalized and separable sobel operators," in *Machine Vision for Three-Dimensional Scenes*, pp. 347–379, Elsevier, 1990.

[17] R. D. Badawi, H. Shi, P. Hu, S. Chen, T. Xu, P. M. Price, Y. Ding, B. A. Spencer, L. Nardo, W. Liu, J. Bao, T. Jones, H. Li, and S. R. Cherry, "First human imaging studies with the EXPLORER total-body PET scanner," *Journal of Nuclear Medicine*, vol. 60, pp. 299–303, Feb. 2019.

[18] M. V. Wickerhauser, *Adapted Wavelet Analysis From Theory to Software*. Nantick, MA., New York: A K Peters/CRC Press, 1994.

[19] I. Daubechies, *Ten Lectures on Wavelets*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1992.

[20] S. Jaffard, Y. Meyer, and R. D. Ryan, *Wavelets: Tools for Science and Technology*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2001.

[21] S. G. Mallat, *A Wavelet Tour of Signal Processing*. Cambridge, MA: Elsevier/Academic Press, 2009.

[22] J. B. J. Fourier, *The Analytical Theory of Heat*. Mineola, NY: Dover Publications, 2003.

[23] E. M. Stein, *Fourier Analysis*. Princeton, NJ: Princeton University Press, 2003.

[24] M. Plancherel and M. Leffler, "Contribution à l'étude de la représentation d'une fonction arbitraire par des intégrales définies," *Rendiconti del Circolo Matematico di Palermo*, vol. 30, pp. 289–335, Dec. 1910.

[25] F. Wechsler, "Noise.jl - adding noise in julia." online, 2021.

[26] J. Irion and N. Saito, "Efficient approximation and denoising of graph signals using the multiscale basis dictionaries," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 3, pp. 607–616, Sept. 2017.

[27] E. Wharton, K. Panetta, and S. Agaian, "Human visual system based similarity metrics," in *2008 IEEE International Conference on Systems, Man and Cybernetics*, IEEE, Oct. 2008.

[28] H. Sheikh and A. Bovik, "Image information and visual quality," *IEEE Transactions on Image Processing*, vol. 15, pp. 430–444, Feb. 2006.

[29] D. Donoho, "De-noising by soft-thresholding," *IEEE Transactions on Information Theory*, vol. 41, pp. 613–627, May 1995.

[30] C. Taswell, "The what, how, and why of wavelet shrinkage denoising," *Computing in Science & Engineering*, vol. 2, pp. 12–19, June 2000.

[31] T. Besard, C. Foket, and B. De Sutter, "Effective extensible programming: Unleashing Julia on GPUs," *IEEE Transactions on Parallel and Distributed Systems*, Sept. 2018.

[32] T. M. Quan and W.-K. Jeong, "A fast discrete wavelet transform using hybrid parallelism on GPUs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, pp. 3088–3100, Nov. 2016.

[33] J. Chen, Z. Ju, C. Hua, B. Ma, C. Chen, L. Qin, and R. Li, "Accelerated implementation of adaptive directional lifting-based discrete wavelet transform on GPU," *Signal Processing: Image Communication*, vol. 28, pp. 1202–1211, Oct. 2013.

Appendix - Supplementary Figures

Figure S1: Relative Error in Denoising when T is m-th Largest Coefficient in Constructed Data

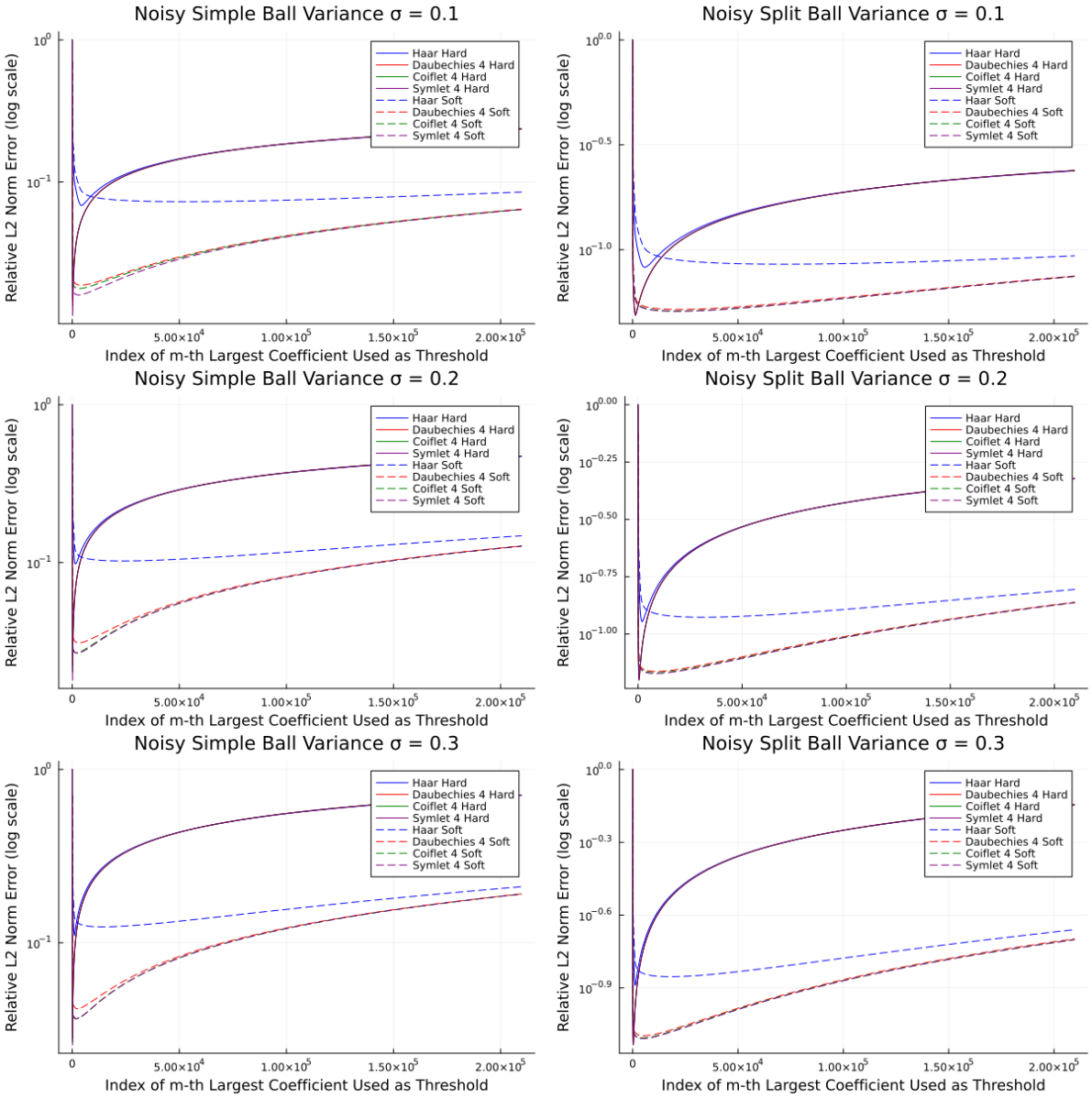


Figure S2: Relative Error in Denoising when T is m-th Largest Coefficient in Medical Datasets