Fast Fourier Transform
vs.
Discrete Wavelet Transform
An Analytic Comparison of Noise Reduction Algorithms

Valerie Bada, Adam Driedger, Tim Wetzel

November 29, 2017

Abstract

Data transforms are used to put data into different formats, in order to manipulate them in ways impossible to achieve in their original forms. These types of transformations are commonly used to "denoise", or reduce noise in, data. This report examines two types of data transforms — the Fast Fourier Transform (FFT) and the Discrete Wavelet Transform (DWT) — and analyzes which is more effective at improving distorted signals. In this report, two signals were tested:

1. Signal 1: a 20-second song clip of 'Gotta Go' by The Incrementals [2], synthetically distorted with noise generated by MATLAB's `rand()` function
2. Signal 2: a seismic signal of a Mw9.0 earthquake originating in Tohoku-Oki, Japan on March 11, 2011 [6], which includes noise organically

Preliminary research indicated that FFT is more accurate when analyzing sound waves due to the periodic nature of the transform, and that DWT is more accurate when analyzing data composed of discrete, unlinked data points. FFT is described as the faster method in all instances. In this report's experiments, this proved true. FFT was measurably faster for both signals, and measurably more accurate for the sound signal. While it was impossible to quantify accuracy against an original seismic signal by the nature of that type of data, DWT was qualitatively more accurate than FFT in this case.

1 Introduction

It is important to begin this topic by explaining the importance of signals and how they are utilized in a modern environment. From music and cell phones to pictures and radars, signals and signal processing are used to transmit important information between sender and receiver in nearly every instance. However, more often than not, these signals become cluttered by interferences in the atmosphere by other signal emitters, called "noise". This is where the idea of noise reduction, or "denoising", comes into play. Denoising is a method of taking an original signal and morphing its properties to provide a more clear picture of where the noise is most affecting the signal. After this, one can remove what is not wanted and morph the signal back, resulting in a cleaner signal. This paper examines two specific signals - a truncated audio file from a Folk-Rock band and a seismographic reading - and analyzes how well two signal processing algorithms reduce noise within each signal. The two algorithms included in this paper are the Fast Fourier Transform (FFT) and the Discrete Wavelet Transform (DWT).

The Fast Fourier Transform is an optimized algorithm for Discrete Fourier Transform (DFT), which takes a signal in the time domain and converts it into the frequency domain. The second

transform, called the Discrete Wavelet Transform, takes the same signal and instead converts it into a coefficient matrix, with stretch and shift coefficients that act on the time variable. As mentioned before, once transformed both algorithms can be used to morph the original signal more easily. This begs the question, which is better? The answer to this is not as cut-and-dry as it may seem. It is more appropriate to ask when one is better than the other, and how it is better. This paper aims to answer these questions.

2 Hypotheses

Since FFT uses a periodic function to decompose signals, and since Signal 1 (like all sound waves) is composed of periodic waves by nature, FFT will be more accurate in denoising Signal 1.

Additionally, because DWT uses a non-periodic function to decompose signals, and since Signal 2 (like all seismic waves) is composed of non-periodic waves by nature, DWT will be more accurate in denoising Signal 2.

Because FFT includes only one transform and inverse transform while DWT includes a series of transforms and an inverse transform, FFT will denoise both Signal 1 and Signal 2 faster than DWT.

3 Model Discussion

3.1 Fast Fourier Transform

The Fast Fourier Transform, as stated previously, is an optimized version of the Discrete Fourier Transform (DFT) that, when broken down to its most basic form, can be represented as the vector-matrix multiplier,

$$\overrightarrow{X} = M * \overrightarrow{x}, \tag{1}$$

with $\overrightarrow{X}$ representing a vector in the frequency domain, $\overrightarrow{x}$ representing a vector of the original time-based signal, and M representing a transformation matrix, defined by

$$M_{kn} = e^{-i2\pi kn/N}, \tag{2}$$

where N is a sequence of complex-valued numbers k [1]. Since the transform is applied to every every element k in N where $0 \leq k \leq N-1$, the series can be approximated by

$$X_k = \sum_{n=0}^{N-1} x_n * M_{kn}. \tag{3}$$

Note that the inverse of this transform looks stinkingly similar to that of original transform,

$$x_n = 1/N \sum_{n=0}^{N-1} X_k * M_{-nk}, \tag{4}$$

where $0 \leq k \leq N-1$ [7]. These equations describe the Discrete Fourier Transform, but when the properties of the odd and even elements of the series and how they behave is considered is when FFT begins to take form. By separating n into the even and odd values, the series can be written as a sum of the two new, re-indexed series

$$X_k = \sum_{n=0}^{(N/2)-1} x_{2m} M_{2mk} + \sum_{n=0}^{(N/2)-1} x_{2m+1} M_{k(2m+1)}, \tag{5}$$

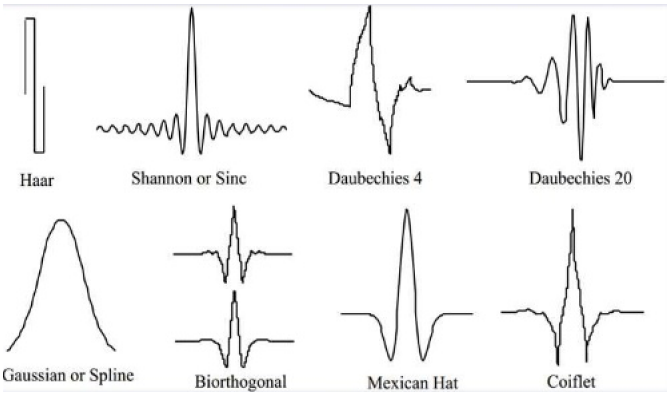


Figure 1: Commonly utilized mother wavelets

with n_{even} reindexing to $2m$ and n_{odd} reindexing to $2m+1$. By plugging in the base function for the matrix M and pulling out the constant term, the equation becomes

$$X_k = \sum_{m=0}^{(N/2)-1} x_{2m} * e^{-i2\pi km/(N/2)} + e^{-i2\pi k/N} \sum_{m=0}^{(N/2)-1} x_{2m+1} * e^{-i2\pi km/(N/2)}. \tag{6}$$

This alone does not save any computational time since the computational cost is still $O[N]$. Notice that this decomposition again has an even value series with in it, the left-hand summation in equation (6), the computation of which has an order $O[N/2]$ [5]. If this is the case, it only makes sense to further decompose the series over and over again. By doing so, the computational cost of the entire summation becomes order $O[NlogN]$, making the Fast Fourier Transform a quicker version of the Discrete Fourier Transform. The same can be said for the inverse as well.

Although FFT denoising functions are not explicitly coded into MATLAB, there are built in functions FFT() and IFFT() which implement these transforms. The MATLAB code for the FFT analysis will be further explained in the "Methods" section.

3.2 Discrete Wavelet Transform

Unlike the Fast Fourier Transform, which uses sine and cosine waves of infinite length to denoise signals, DWT utilizes wavelets, which are finite in length and scaled from a chosen mother wavelet. The process of DWT consists of choosing a mother wavelet, decomposing a signal into low pass and high pass filters, thresholding to obtain the desired coefficients for the function, and reconstructing the signal to get the desired "denoised" signal.

The process of DWT begins with the selection of a mother wavelet. This mother wavelet will essentially slide along the time domain and be scaled and shifted in order to closely approximate the original data. Mother wavelets are characterized by properties such as orthogonality, compact support, symmetry, and vanishing moment [11]. Different mother wavelets applied on the same signal may produce different results, so this selection is important. There are many different mother wavelets to choose from (Figure 1).

The chosen mother wavelet will henceforth be denoted as $\psi_{j,k}(x)$. Then every signal can be represented by the series

$$f_{j+1}(x) = f_j(x) + \sum_{k=0}^{2^j-1} d_{j,k}\psi_{j,k}(x) = \sum_{k=0}^{2^j-1} c_{j,k}\psi_{j,k}(x) + \sum_{k=0}^{2^j-1} d_{j,k}\psi_{j,k}(x), \tag{7}$$

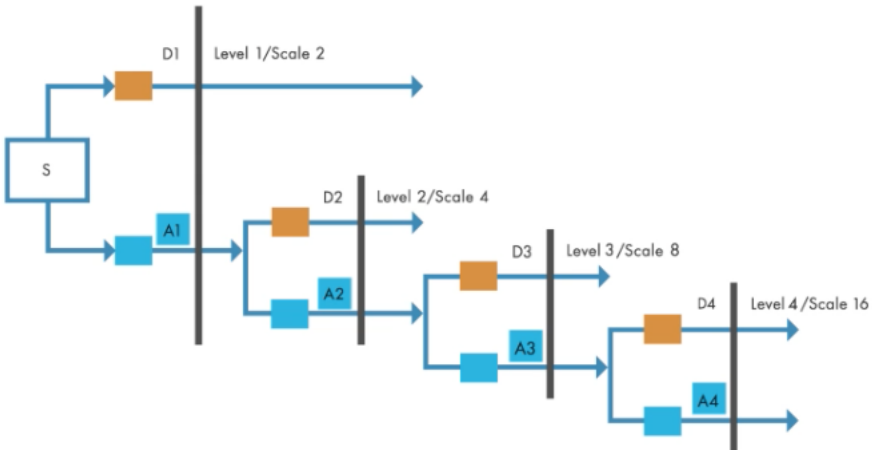
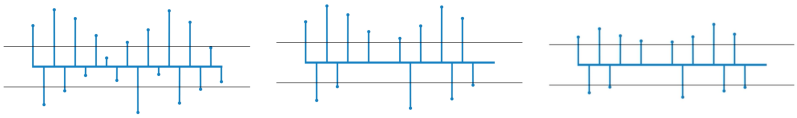


Figure 2: Decomposition of a data set using high-pass and low-pass filters



(a) A set of coefficients (blue) and the set threshold value (grey)

(b) Signal 1, with noise

(c) Coefficients less than or equal to threshold value set to zero

Figure 3: Coefficients greater than threshold value shrunk towards zero

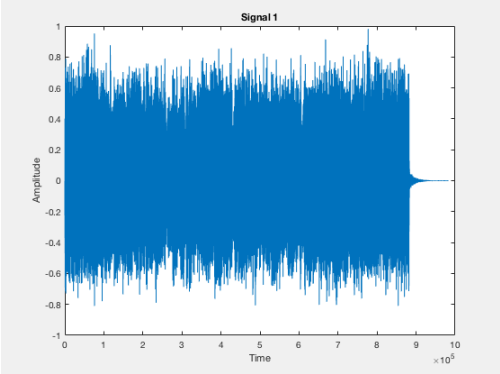
where $c_{j,k}$ is the set of coarse detail coefficients and $d_{j,k}$ is the set of fine detail coefficients [10].

A method of wavelet decomposition is the first step to obtaining these coefficients. The chosen mother wavelet determines a low-pass filter, `Lo_D`, and a high-pass filter, `Hi_D`, corresponding to the low frequencies and the high frequencies respectively. The Discrete Wavelet Transform function implemented by MATLAB takes in the original signal and the decomposition filters and produces an approximation coefficient vector A and a detailed coefficient vector D. In Figure 6, notice that this separation of the signal's information is iterated a number of times. The number of iterations is equivalent to the level of the decomposition.

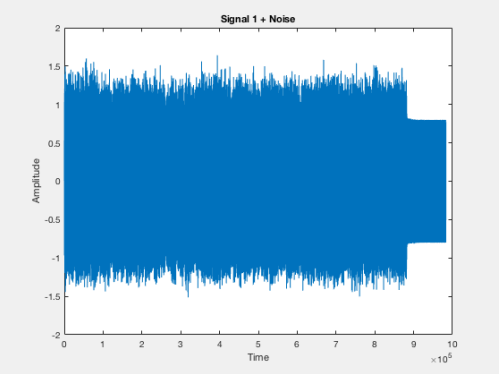
Now that the signal has been decomposed, filtering out unwanted noise can be achieved through the process of thresholding. Thresholding essentially creates a region around zero where the coefficients are considered negligible. Figure 3 illustrates the process of soft thresholding in particular. In this process, the coefficients obtained from the decomposition process are compared to a specified threshold coefficient. Each data point less than or equal to the specified threshold value is then set to zero. Every coefficients greater than the threshold value is then shrunk towards zero by a factor proportional to its original distance from the threshold value. The process of hard thresholding is equivalent until the third step, where all coefficients greater than the threshold value are set to equal the threshold value. [12]

With the negligible coefficients removed from the data set, the signal is now reconstructed using a process that inverts MATLAB's wave decomposition function. If the most ideal parameters were chosen, then the produced signal will be denoised.

The wavelets obtained this way have special scaling properties. They are localized in time and frequency, permitting a closer connection between the function being represented and their

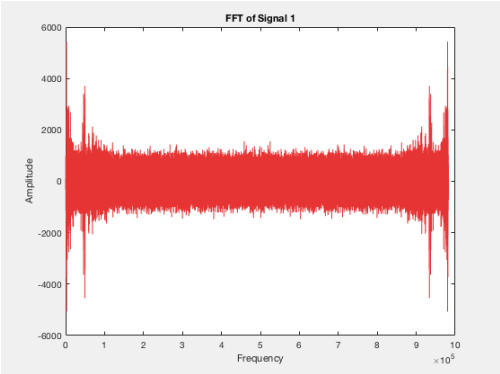


(a) Signal 1

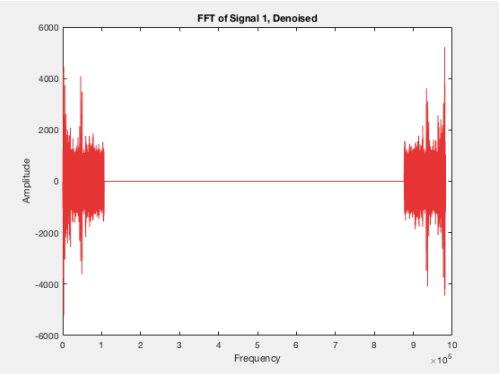


(b) Signal 1, with noise

Figure 4: Signal 1, with and without noise



(a) FFT of Signal 1



(b) FFT of Signal 1, denoised

Figure 5: FFT of Signal 1, with and without noise

coefficients. This makes DWT ideal for denoising signals with extreme local details such as sudden peaks and discontinuities.

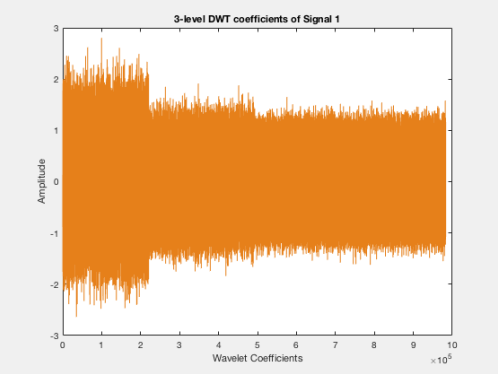
4 Results

4.1 Signal 1: Song Clip

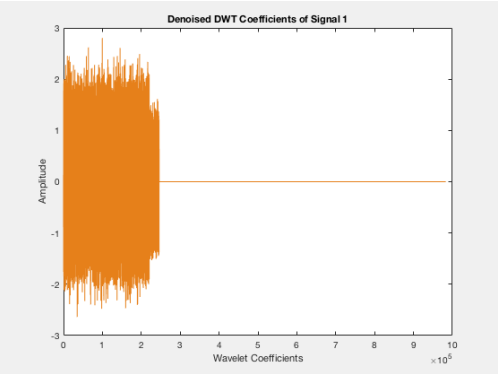
Signal 1 was a 20 second sound clip of the song 'Gotta Go,' written and performed by The Incrementals [2] (Figure 4(a)). A manipulation of the `rand()` function created a 20-second noise clip which distorted the sound file to prepare it for the test (Figure 4(b)).

4.1.1 FFT on Signal 1

To prepare for denoising, the noisy signal was first transformed using FFT into the frequency domain. This allowed for easier identification of abnormal noise, as there was an obvious point on the frequency domain at which the majority of the signal was random noise (Figure 5(a)). This noise was eliminated by setting the transformed data equal to 0 after a threshold point, which was relatively obvious in this case as well due to its difference in uniformity from the rest of the data (Figure 5(b)).

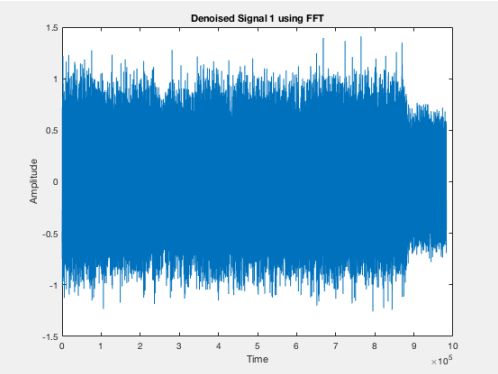


(a) DWT of Signal 1

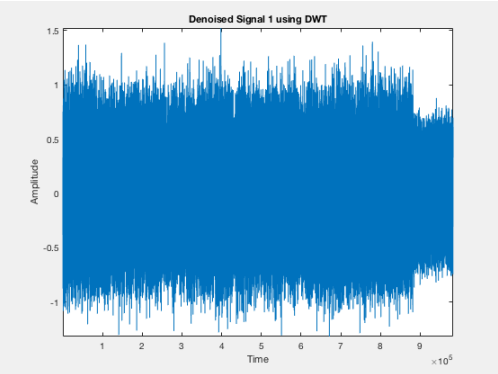


(b) DWT of Signal 1, denoised

Figure 6: DWT of Signal 1, with and without noise



(a) Signal 1, denoised using FFT



(b) Signal 1, denoised using DWT

Figure 7: Comparison of denoised Signal 1

After the signal was denoised, it was transformed back to its original time domain using `MATLAB`'s IFFT() function. This yielded a signal denoised through FFT.

4.1.2 DWT on Signal 1

The Discrete Wavelet Transform code was implemented in a way extremely similar to FFT. First, the noise was added identically to the FFT code. Next, using MATLAB's `dwt()` function, the data was transformed into a matrix of coefficients to separate out abnormal-frequency noise from the rest of the signal (Figure 6(a)). This was done using a 3-level DWT (i.e., running the transform 3 times and obtaining 5 sets of "detail" coefficients), as this did the best job of denoising the signal without eliminating necessary data from it. After this, the "fine detail" coefficients, which contained a good amount of the white noise, were discarded (Figure 6(b)).

With the fine detail coefficients at 0, the entire matrix was transformed back to its original domain using `MATLAB`'s IDWT() function. This yielded another signal, this time denoised through DWT (Figure 7(b)).

4.2 Signal 2: Seismic Data

Next, Signal 2 a seismic signal of a Mw9.0 earthquake originating in Tohoku-Oki, Japan on March 11, 2011 (Figure 8 went through the exact same process as Signal 1, minus the step to add noise

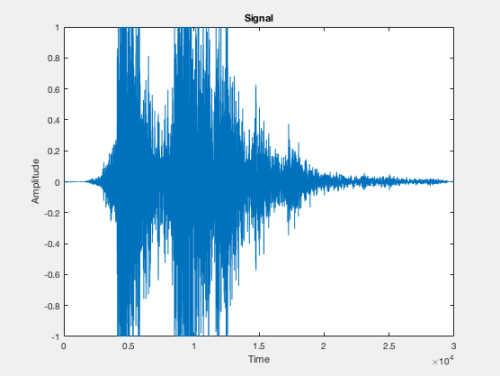
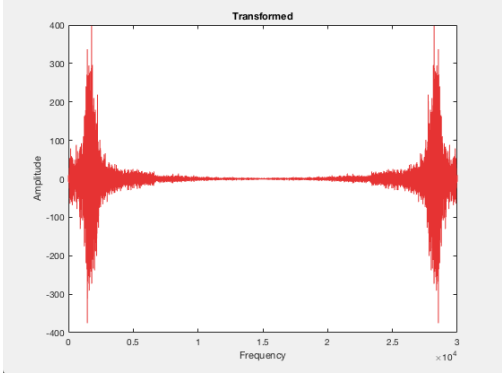
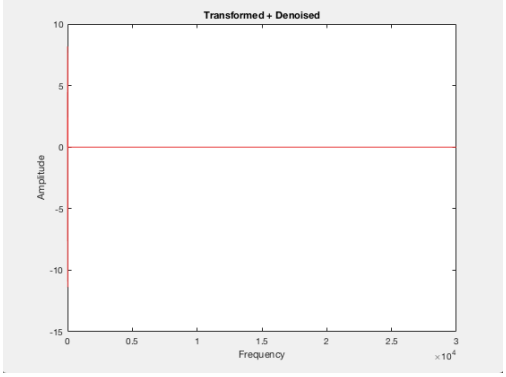


Figure 8: Signal 2



(a) FFT of Signal 2



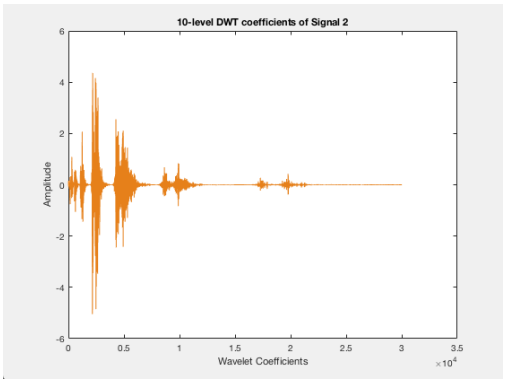
(b) FFT of Signal 2, denoised

Figure 9: FFT of Signal 2, with and without noise

to the signal (since seismic data is inherently noisy).

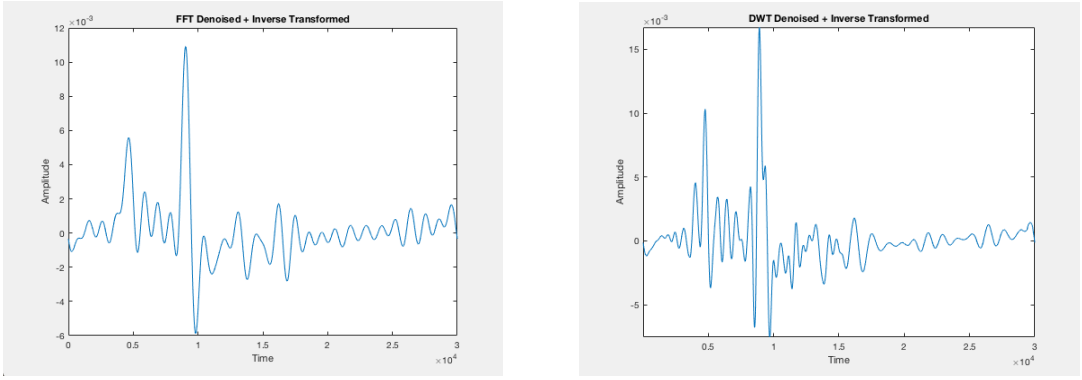
4.2.1 FFT on Signal 2

Similar to Signal 1, Signal 2 was first transformed into the frequency domain (Figure 9(a)). Next, it was denoised. It's worth noting that our threshold for this signal was much lower, which took out a huge amount of the data (Figure 9(b)). This was done to isolate the major signal changes, which correspond to the initial pressure waves (p waves) and shear waves (s waves) in the earthquake the largest two waves, and the most significant in preliminary analysis of seismic data [3]. Last, the data was transformed back into the time domain, giving us a much simpler signal, denoised using FFT (Figure 11(a)).



(a) DWT of Signal 2 (b) DWT of Signal 2, denoised

Figure 10: Comparison of denoised signals



(a) Signal 2, denoised using FFT (b) Signal 2, denoised using DWT

Figure 11: Comparison of denoised signals

4.2.2 DWT on Signal 2

DWT on Signal 2 was also carried out in a way extremely similar to Signal 1. First, the signal was transformed into its coefficients (Figure 10(a)). Next, a 10-level version of the DWT denoising code eliminated a huge number of coefficients, leaving only a very small number to isolate the p and s waves in the signal (Figure 10(b)).

After its transform was denoised, the signal was inversely transformed back into the original time domain, using `MATLAB`'s IDWT() function again (Figure 11(b)). This yielded a new signal denoised using DWT.

5 Results

The results of the tests confirm the hypotheses. As outlined below, FFT was more computationally efficient in both cases. In addition, it was more accurate in denoising the song clip, which is composed of many periodic waves. DWT, on the other hand, was more accurate in the denoising of seismic data, which is composed largely of wavelets.

5.1 Signal 1: Song Clip

To compute the average computational run time of FFT, all extraneous code (plots, sounds, etc.) was deleted from the code. Using a for loop and MATLAB's tic() and toc() functions, the code was iterated 100 times. The averages below are averages of those 100 iterations.

Average Computational Run Time (Signal 1):
FFT time: 0.2473 seconds
DWT time: 0.4955 seconds

As shown above, the DWT denoising code took nearly twice as long on average as the FFT denoising code. Next, MATLAB's corrcoef() function computed a correlation coefficient between the original Signal 1 and each denoised signal, to see how well each denoised signal compared to the original. A perfect match would be indicated by a 1.0000, while no similarity would be denoted by 0.0000.

Correlation Coefficient (Signal 1):
FFT correlation: 0.6783
DWT correlation: 0.6658

The accuracy of the two algorithms can be seen above. Both recaptured a good amount of Signal 1, but FFT still slightly outperformed DWT in accuracy. It's possible that this difference would be more drastic with a larger signal.

5.2 Signal 2: Seismic Data

Average Computational Run Time (Signal 2):
FFT time: 0.0080 seconds
DWT time: 0.0180 seconds

As hypothesized, on Signal, 2 FFT outperformed DWT on run time by 0.01 seconds. However, it was impossible to compute a correlation coefficient between the denoised and original signals, since no original signal exists for seismic data. This is because for seismic data, the original "signal" is simply seismographs picking up noisy earthquake data at different points. However, Professor Zhigang Peng, who teaches geophysics at Georgia Institute of Technology, also performed a noise reduction algorithm on the same seismic signal. Side-by-side comparisons of his data and these two denoised signals shows that DWT is slightly more accurate in its seismic denoising than FFT, as the DWT code picks up more of the nuances of the signal without leaving in unnecessary bumps (Figures 12, 13, 14).

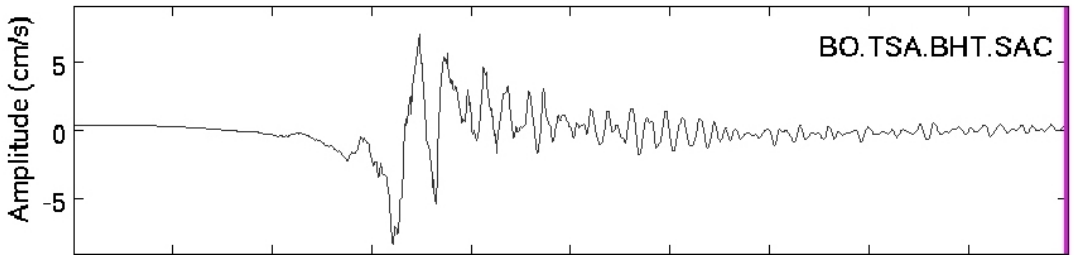


Figure 12: Signal 2, collected by the F-Net station TSA and interpreted by Dr. Zhigang Peng

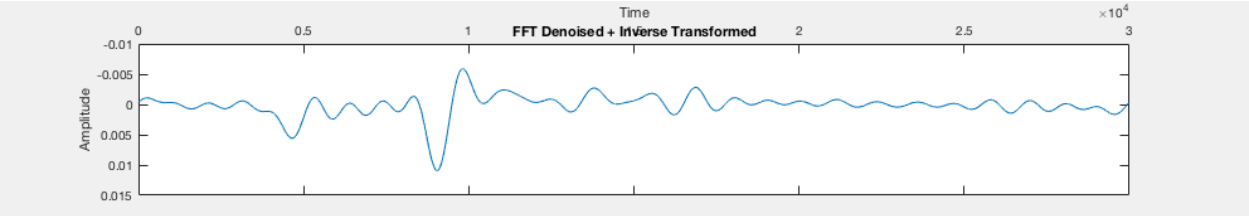


Figure 13: Signal 2, denoised using FFT

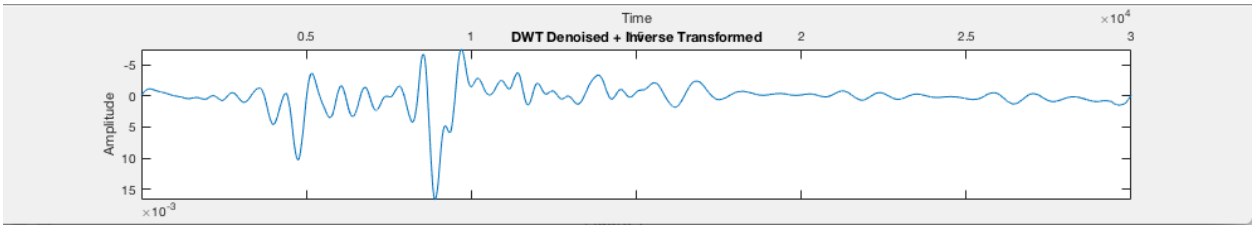


Figure 14: Signal 2, denoised using DWT

5.3 Critique

As shown above, the results of the analysis are based on the MATLAB code which included some fundamental assumptions that can result in some deviation from the true result. However, it is important to point out that these assumptions were applied to both algorithms, and so they should theoretically differ to the same degree. When a threshold was decided for the transformed signals, the point at which all proceeding noise signal was floored or set to zero was picked based on the visual representation of the signal. Because this point was picked by trial-and-error rather than systematically, there was likely a small degree of inaccuracy in the resulting signal. Some of the original signal was most likely lost in this process, or some of the noise was left in. This was especially problematic in Signal 2, where the threshold was so small that it was picked experimentally at first to test how different recomposed data looked after denoising. Additionally, with no quantitative way to compare the denoised versions of Signal 2, the selection of DWT as more effective is highly subjective.

Another problem with these models was the noise added to Signal 1. To make the transforms more effective in denoising the signal, only randomly-generated noise was added. Adding more realistic noise, such as an audio file from a crowded room, proved extremely difficult to filter out using either transform. Low-frequency noise gets stuck in the original

data, and when using either of these transforms, remains near impossible to filter out without removing copious amounts of the original signal. Therefore, high-frequency noise was originally added to the signal to demonstrate more accurately the differences between the two transforms; however, both transforms were actually too effective at denoising this signal, with both yielding a 100% correlation to the original. Therefore, completely random noise was added instead to more accurately demonstrate the difference in quality between the two transforms, but both transforms denoised the signal less accurately as a result.

6 Summary

In summation, the analytic comparison of the Fast Fourier Transform and Discrete Wavelet Transform confirmed the hypothesized results. The code using FFT denoised Signal 1 more quickly and accurately than that using DWT. On the other hand, while the code using FFT also denoised Signal 1 more quickly, the code using DWT denoised it more accurately. Therefore, in these tests, the hypotheses held.

References

[1] Brigham, E. Oran. "The Fast Fourier transform." Englewood Cliffs: Prentice-Hall (1974).

[2] The Incrementals. Gotta Go. Soundcloud.com. Rec. 2015. MP3.

[3] Jeans, J. H. The Propagation of Earthquake Waves. Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character, vol. 102, no. 718, 1923, pp. 554574. JSTOR, JSTOR, www.jstor.org/stable/94031

[4] MATLAB. MATLAB:2017, Version 9.3 (R2017b),The MathWorks Inc., Natick, Massachusetts, 2017

[5] Thomas, Millicent, et al. "Seismic signal analysis using multi-scale/multi-resolution transformations." Applied Imagery Pattern Recognition Workshop (AIPR), 2016 IEEE. IEEE, 2016.

[6] Peng, Zhigang, Earthquake Sound of the Mw9.0 Tohoku-Oki, Japan earthquake (Zhigang Peng), Wed. 7 March 2012 08:47:34

[7] Sauer, T. Numerical Analysis 2e. *Pearson*, pp. 178–187, 1978

[8] Verschelde, Jay. MATLAB Lecture 7. Signal Processing in MATLAB, UIC, Dept of Math, Stat & CS, 23 April 2007

[9] Wu, Ja-Ling. Fast Fourier Transform (FFT), National Taiwan University.

[10] Wu, Yi-Leh, Divyakant Agrawal, and Amr El Abbadi."A comparison of DFT and DWT based similarity search in time-series databases." Proceedings of the ninth international conference on Information and knowledge management. ACM, 2000.

[11] Ngui, Wai Keng, et al. "Wavelet analysis: Mother wavelet selection methods." Applied mechanics and materials. Vol. 393. Trans Tech Publications, 2013.

[12] Chang, S. Grace, Bin Yu, and Martin Vetterli. "Adaptive wavelet thresholding for image denoising and compression." IEEE transactions on image processing 9.9 (2000): 1532-1546.

[13] Lee, Daniel TL, and Akio Yamamoto. "Wavelet Analysis: Theory and Applications." Hewlett Packard journal 45 (1994): 44-44.