APPLICATIONS IN TIME-FREQUENCY DOMAIN ANALYSIS

APPLICATIONS IN TIME-FREQUENCY DOMAIN ANALYSIS

BY

VINAY YUVASHANKAR, B.Eng.

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTING AND SOFTWARE

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

Master of Applied Science (2017) McMaster University

(Computing and Software) Hamilton, Ontario, Canada

TITLE:	APPLICATIONS IN TIME-FREQUENCY DOMAIN ANALYSIS
AUTHOR:	Vinay Yuvashankar B.Eng. (Mechatronics) McMaster University, Hamilton, Canada
SUPERVISOR:	Dr. Martin Von Mohrenschildt
NUMBER OF PAGES:	xviii, 85

To the people who challenged my view of the world

Abstract

Time-Frequency decomposition is a signal processing method for analyzing and extracting information from aperiodic signals. Analysis of these signals are ineffective when done using the Fourier transform, instead these signals must be analyzed in the time and frequency domain simultaneously. The current tools for Time-Frequency analysis are either proprietary or computationally expensive making it prohibitive for researchers to use. This thesis investigates the computational aspects of signal processing with a focus on Time-Frequency analysis using wavelets. We develop algorithms that compute and plot the Time-Frequency decomposition automatically, and implement them in C++ as a framework. As a result our framework is significantly faster than MATLAB, and can be easily incorporated into applications that require Time-Frequency analysis. The framework is applied to identify the Event Related Spectral Perturbation of EEG signals; and to vibrational analysis by identifying the mechanical modal parameters of oscillating machines.

Acknowledgments

I would like to acknowledge Dr. Martin Von Mohrenschildt, this thesis took 29 revisions to complete, thank you for your honest feedback.

I would also like to acknowledge Dr. Madiha F. Khan, who provided valuable feedback and support during the writing process.

Notation and abbreviations

$x[\cdot]$	A function in the discrete time domain
$x(\cdot)$	A function in the continuous time domain
$\hat{x}(\cdot)$	The Fourier Transform of the function
$\mathcal{P}(\cdot)$	Energy Spectrum Function
$\mathcal{W}(\cdot)$	Continuous Wavelet Transform Function
$\Psi(\cdot)$	Wavelet Function
BDF	BioSemi Data Format
CTFT	Continuous Time Fourier Transform
CWT	Continuous Wavelet Transform
DFT	Discrete Fourier Transform
DTFT	Discrete Time Fourier Transform
EDF	European Data Form
EEG	Electroencephalography
ERP	Event Related Potential
ESRP	Event Related Spectral Perturbation
FFT	Fast Fourier Transform
FFTW	Fastest Fourier Transform in the West

GSL	GNU Scientific Library
PNG	Portable Network Graphics
SIMD	Single Instruction, Multiple Data
STFT	Short Time Fourier Transform

Contents

List of Tables

List of Figures

Chapter 1

Introduction

A signal is a function of time that can convey information [43]. It can originate from a variety of sources (e.g. deep space electromagnetic radiation vs. orchestral sounds) [44]. In the physical world a signal is analog, but usually it is converted and processed digitally. Analog signals exist in the continuous time domain where the changes in amplitude are tracked over time. Digital signals exist in the discrete time domain, where the function is represented at regular intervals by a finite set of values. The desire to study signal processing methods is fueled by the recent improvements in computational speed and the concomitant cost-reduction of sensors.

One of the goals of signal processing is to identify repeating patterns that occur periodically. The amount of time it takes for a signal to repeat is known as its period, and the amount of repetitions that occur within a second is known as its frequency. These patterns can be analyzed with the Discrete Fourier Transform (DFT), a mathematical method that decomposes a periodic signal into weighted complex exponentials.

However, this method is ineffective at analyzing aperiodic signals. A signal, such as

the sounds from an orchestra, has information in the temporal domain that would be lost in the decomposition. By decomposing a signal in terms of complex exponentials, which are periodic functions, the DFT removes the temporal information. Methods to analyze a signal's frequency and temporal information simultaneously — known as Time-Frequency analysis — are required. In this document we will examine two Time-Frequency methods called the Short Time Fourier Transform (STFT), and the Continuous Wavelet Transform (CWT).

The STFT modifies traditional Fourier analysis by analyzing the signal in smaller equally sized segments called windows. By successively performing the Fourier transform in smaller segments we can gain some temporal resolution. It can be seen that the temporal resolution is determined by the size of these windows; smaller windows are better equipped to localize a signal's attributes in time. However, dividing the signal comes with a cost, it trades temporal resolution for frequency. The smaller windows would be unable to analyze periods that occur over a long time because the window is not large enough to encompass it. The window size could be increased to identify the slow changes, but this would in turn reduce the resolution in the time domain.

The CWT improves on the STFT by adjusting the window size according to the frequency. This is accomplished by its use of wavelets, rather than complex exponentials, as its analysis function. Wavelets have finite energy localized at a point in time, while complex exponentials have energy at all times. Wavelets with the same energy can be expanded for large windows, and compressed for small windows. The signal is then correlated with both large and small windows which provides us with a detailed decomposition of the signal's entire frequency spectrum as a function of

time.

While the STFT and CWT are powerful tools, they are expensive to compute for non-trivial signals. This thesis discusses methods of generating the Time-Frequency decompositions automatically. It begins with a short review of the mathematics of the Fourier transform and introduces the need for Time-Frequency analysis. It then describes the STFT and CWT as two methods for decomposing aperiodic signals. The developed algorithms numerically approximate these methods, and are implemented as a framework in the C and C++ programming languages. The framework is applied to two signal processing scenarios in which Time-Frequency analysis is required. First, it is compared with standard packages available for neuroscience researchers who work with Electroencephalography (EEG) analysis. Second, due to work done within our research group, the framework is applied to identify the mechanical modal parameters of an oscillating machine.

Chapter 2

Background

The focus of this chapter is to provide the background mathematics that will be used to analyze a signal in the Time-Frequency domain. We will first review the time and frequency domains, and introduce the Fourier Transform as a method of analyzing a signal's frequency information. Time-Frequency analysis is then introduced to overcome the limitations of Fourier analysis by analyzing a signal in both time and frequency. We then discuss the Short Time Fourier Transform (STFT) and the Continuous Wavelet Transform (CWT) as two methods that can decompose a signal in the Time-Frequency domain.

2.1 The Time and Frequency Domains

A signal in the temporal domain shows the general nature of its changes as a function of time, i.e. whether it is sinusoidal, random, repetitive, or transient in nature [53]. A signal such as a cosine function, which has a natural frequency of ω_0, can be represented in the time domain as

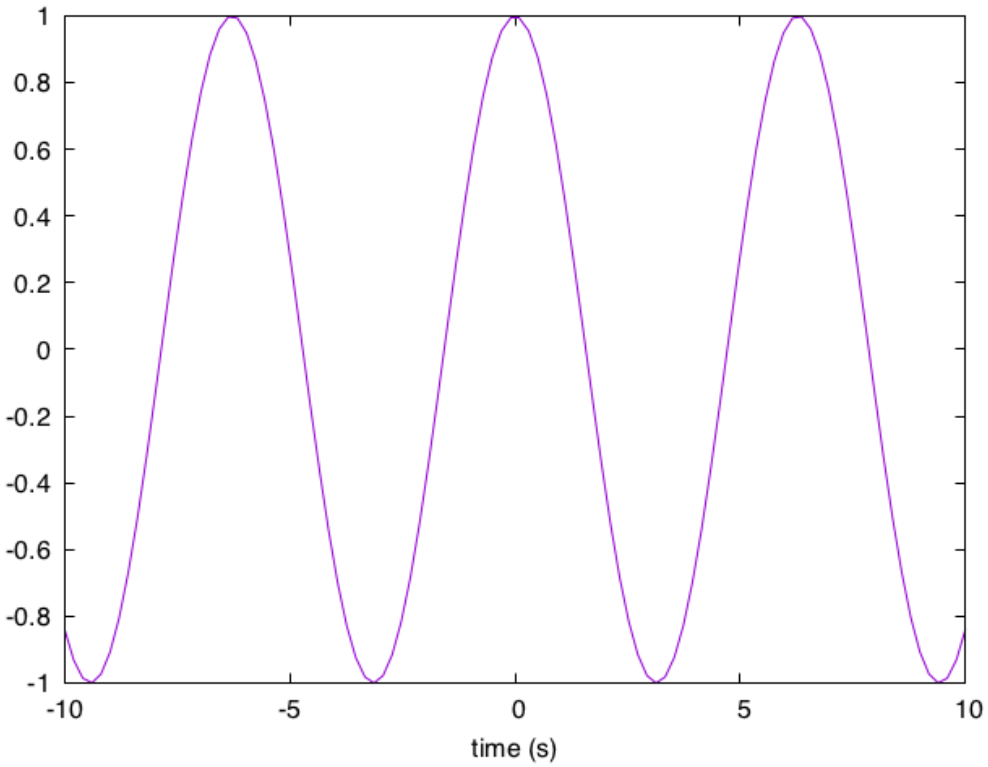


Figure 2.1: The time representation of a cosine function

$$x(t) = cos(2\pi\omega_0 t). \tag{2.1}$$

The cosine signal, visualized in Figure 2.1, repeats its values periodically. The time it takes for the function to repeat or cycle is known as its period, given by p, measured in seconds. Its frequency is the number of cycles it undergoes in one second, defined as the inverse of the period, or

$$f = \frac{1}{p}$$

and is measured in Hertz. A signal in the time domain can be transformed into

the frequency domain by separating it into weighted complex exponentials called the energy spectrum density [27].

A periodic cosine function, such as equation 2.1, is represented in the frequency domain as

$$X(\omega) = \frac{1}{2}\delta(\omega - \omega_0) + \frac{1}{2}\delta(\omega + \omega_0).$$

The energy spectrum density of the cosine function, visualized in Figure 2.2, is defined by two Dirac delta functions at the cosine's natural frequency ω_0. The Dirac delta function is given by

$$\delta(t) = \begin{cases} +\infty & t = 0 \\ 0 & t \neq 0 \end{cases}, \tag{2.2}$$

where it is constrained by

$$\int_{-\infty}^{+\infty} \delta(t)dt = 1. \tag{2.3}$$

The frequency domain shows the energy and the phase shift of a signal as a function of angular frequency ω. It is often used to identify the periodic properties that compose a signal.

Aperiodic signals can be analyzed in the frequency domain as well. For example, an aperiodic impulse function

$$x(t) = \delta(t), \tag{2.4}$$

is represented as a Dirac delta, would have an energy spectrum density of

$$X(\omega) = 1. \tag{2.5}$$

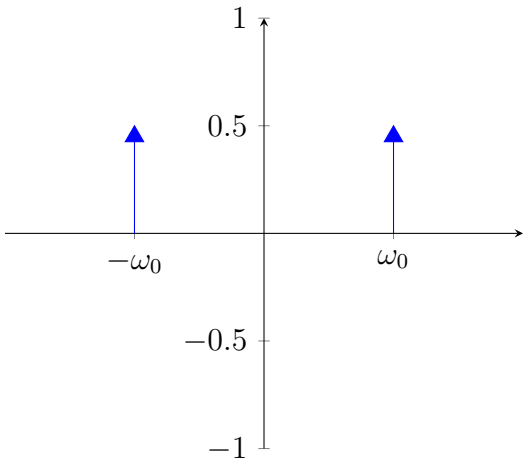
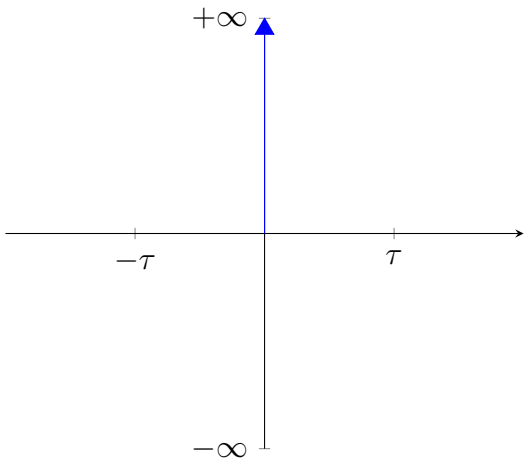


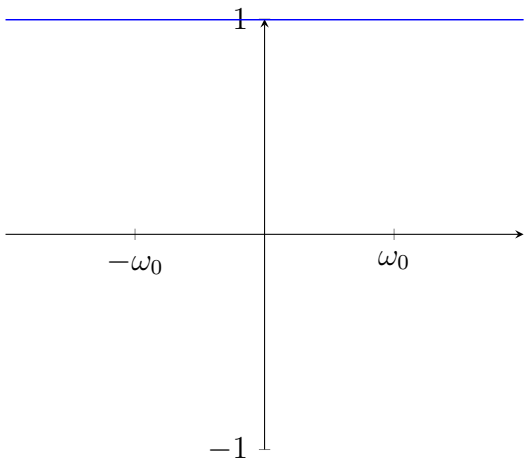
Figure 2.2: The Energy Spectrum of a Cosine Function

These functions are visualized in Figure 2.3. Note that the transformation of the signal from the time domain to the frequency domain changes its nature. The continuous signal is transformed into a discrete function, and the discrete function is transformed into a continuous signal. This occurs when transforming a signal from one domain to another.

The time and frequency domains highlight different properties in a signal. The frequency domain shows the periodic and cyclic properties; while the time domain is useful for identifying instantaneous and transient vibrations. Both domains are important in signal processing as one provides information the other does not. For example, an impulse function and random noise would have identical energy spectrum densities, it is only in the temporal domain that their transient nature would be distinguished. Conversely, one would not use the temporal domain to analyze a signal's periodic properties, it would be easier to transform the signal into the frequency domain to identify these features. Transforming signals to and from the time and frequency domains is done through a method called Fourier analysis.



(a) An Impulse Function



(b) The Energy Spectrum Density

Figure 2.3: Figure 2.3a shows the Impulse Function, and Figure 2.3b shows the Energy Spectrum Density of the Impulse Function

2.2 The Fourier Transform

Fourier analysis decomposes a signal into complex exponentials [6]. The Continuous Time Fourier Transform (CTFT) is defined as

$$X(\omega) \approx \int_{-\infty}^{+\infty} x(t)e^{-i\omega t}dt,$$

where: i is the complex number equal to $\sqrt{-1}$, $x(t)$ is a continuous signal for which an integral exists, and ω is the angular frequency measured in radians per second. The periodic nature of the complex exponential in the Fourier transform implies the function $X(\omega)$ has a period of 2π. This property is defined as

$$X(\omega) = X(\omega + 2k\pi),$$

where k is any integer. The CTFT can transform any aperiodic temporal signal into an aperiodic function in the frequency domain.

The CTFT can be discretely sampled as the Discrete Time Fourier Transform (DTFT) given by

$$X(\omega) \approx \sum_{n=-\infty}^{+\infty} x[n]e^{-i\omega n},$$

where $x[n]$ is a discrete and aperiodic function, and n can be any integer. The DTFT definition is valid if it is assumed that the values of x and ω allow the infinite sum to converge. The DTFT transforms a signal from the aperiodic time domain into the periodic frequency domain. However, if the signal is discrete and periodic, then the Discrete Fourier Transform (DFT) can be used.

The DFT is given as

$$X_k \approx \sum_{n=0}^{p-1} x[n]e^{-i\omega_0 kp}, \tag{2.6}$$

where: p is an integer representing the period, X_k are the Fourier series coefficients, ω_0 is defined as $2\pi/p$, and k is any integer. The DFT transforms a periodic time domain signal into a periodic function in the frequency domain. It uses a complex

exponential in its analysis method which means it is Np periodic, or

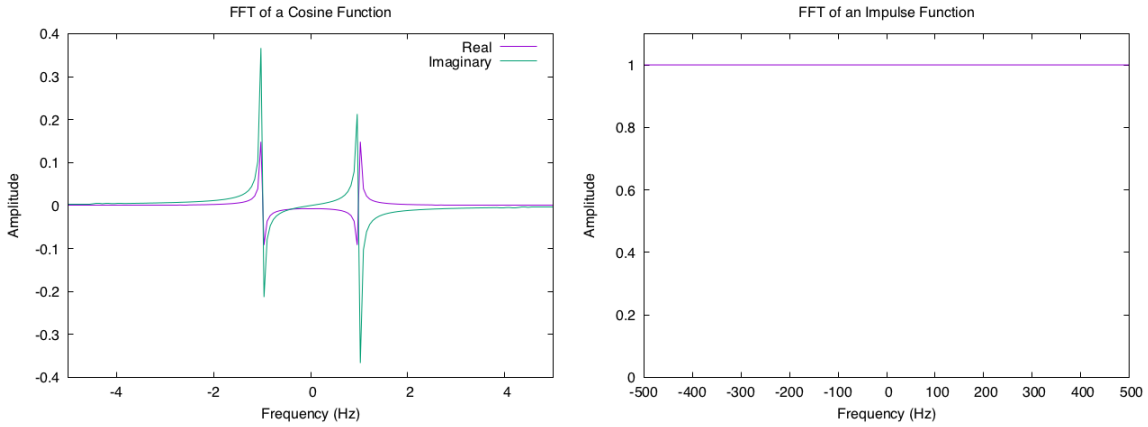
$$X_k = X_{k+Np},$$

where N and k can be any integer. A discrete and periodic signal can be analyzed and transformed into the frequency domain by using equation 2.6. It can conversely be synthesized with the Inverse Discrete Fourier Transform (IDFT) which is given as

$$x[n] = \frac{1}{p} \sum_{n=0}^{p-1} X_k e^{i\omega_0 n}.$$

The DFT is often used in signal processing because both the forward and inverse transforms are discrete and finite. Their discrete nature makes them ideal for digital circuits, and their finite summations ensure that a solution exists. The transformations between the time and frequency domains can be made less expensive by automatic, rather than analytic, calculations.

The forward and inverse Discrete Fourier Transforms are computationally expensive in the order of $\mathcal{O}(n^2)$. In practice, a Fast Fourier Transform algorithm — such as the Cooley-Tukey algorithm — is used to compute the DFT and IDFT, which reduces the computational complexity to $\mathcal{O}(n \log(n))$ [11]. The Fast Fourier Transform (FFT) achieves this speed by numerically approximating the results of the DFT. Figures 2.4a, and 2.4b are the FFT of the cosine and the impulse functions discussed in Section 2.1. It is evident that the numerical approximations do not match the energy spectrums found in the previous section, but they are accurate enough to identify the frequency information of a signal.



(a) Cosine Function (b) Impulse Function

Figure 2.4: This figure shows the FFT of the cosine function (left) and the impulse function (right), both functions were sampled at 512 Hz for 8000 samples. The cosine function's natural frequency was 1.0 Hz, and the impulse function was computed with a Kronecker delta function.[1]

The FFT completely transforms a function from the time domain into the frequency domain. This process loses all temporal information which renders the processing of some signals more difficult. As it was pointed out earlier, the energy spectrum densities of an impulse and random noise are the same. The impulse is localized in time, while the random noise exists at all times. Since the time information is lost in the analysis process, it is not possible to differentiate between these signals in the frequency domain.

2.3 Time-Frequency Analysis

Time-Frequency analysis is the method for studying a signal in both the temporal and frequency domains. Fourier analysis is a complete transformation of a signal into the frequency domain, which is ineffective for analyzing aperiodic signals. These

[1]The Kronecker Delta Function is defined in Section 6.1

signals have information in the time domain that are best described in both time and frequency (see Figure 2.5).

The mathematics of Time-Frequency analysis imposes a limit to the precision of localizing a signal in time and frequency. The frequencies that compose a signal is defined in terms of its oscillations versus the length of time. A signal needs time to oscillate at a certain frequency, therefore it does not simultaneously have a precise frequency at a certain time [17].

This phenomenon is best described by the Heisenberg Uncertainty Principle. In quantum mechanics, one cannot accurately know a particle's position and momentum simultaneously. When a particle's position is known with a high probability, its momentum is less certain; and if the particle's momentum is more certain, then its location is not certain [47]. There is a trade-off between these two domains, and Time-Frequency analysis is working with the trade-off between the time and frequency domains. The Time-Frequency methods discussed in this thesis takes a one-dimensional temporal signal and represents it in a two-dimensional function of both time and frequency [42].

2.3.1 Short Time Fourier Transform

One method of exploring the trade off between the temporal and frequency resolutions is the Short Time Fourier Transform or STFT. The STFT divides a signal into shorter windows of equal length and then performs the DFT for each window [31]. The STFT is defined as

$$STFT(\tau, \omega) = \int_{-\infty}^{+\infty} x(t)h(t-\tau)e^{-i\omega t}dt. \tag{2.7}$$

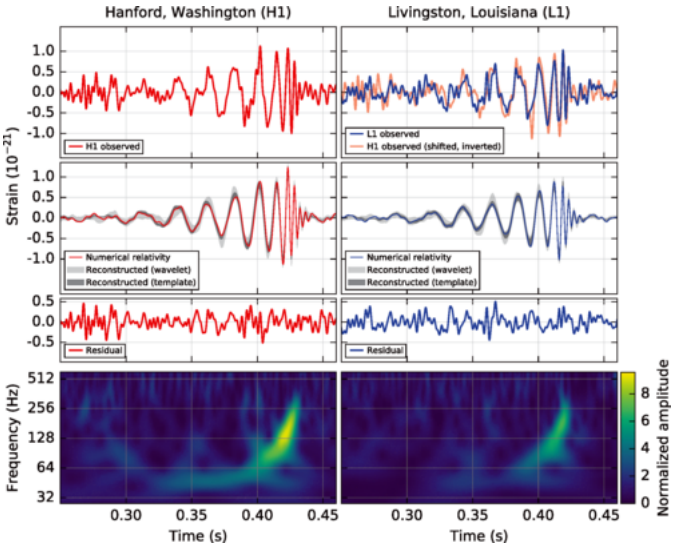


Figure 2.5: Time-Frequency Analysis was used to identify the gravitational waves emitted from two merging black holes. The key identifying feature of these gravitational waves is that the frequency increases at a very particular rate with respect to time. Time-Frequency analysis is required to classify these signals accurately [1].

(a) STFT at a high temporal resolution (b) STFT at a low temporal resolution

Figure 2.6: The STFT of a signal at different temporal resolutions

A window function $h(t)$ is multiplied with the signal $x(t)$, and then analyzed using the Fourier transform. The window function leaves the signal unaltered around time τ and reduces anything distant from time τ to zero [10]. One would usually use a Rectangular or Hann Window function for the STFT of a signal. The result is complex but typically it is plotted as the changing power spectrum as a function of time, which is given as

$$\mathcal{P}(\tau, \omega) = |STFT(\tau, \omega)|^2. \tag{2.8}$$

The STFT introduces the trade-off with the time and frequency domains by shortening the DFT windows — adding temporal information to the analysis process. There are limitations however, one could shorten the windows to improve the temporal resolution, but this means that the frequency resolution suffers. The shorter windows would be unable to analyze the lower frequency components of the signal. While on the other hand, if the window is kept large, then the frequency's temporal location is less accurate. To gain resolution in one domain, one must lose resolution in the other. This is because the same Time-Frequency window that is used in the lower frequencies are also used in the higher frequencies (see Figure 2.7). These non-adaptive windows are either too large or too short to analyze the variety of frequencies that can exist in a signal [52]. Figure 2.6 shows the STFT conducted on the same signal with two different window sizes. Wavelet analysis improves on the STFT to provide a better trade-off between the Time-Frequency resolution by changing the time and frequency windows.

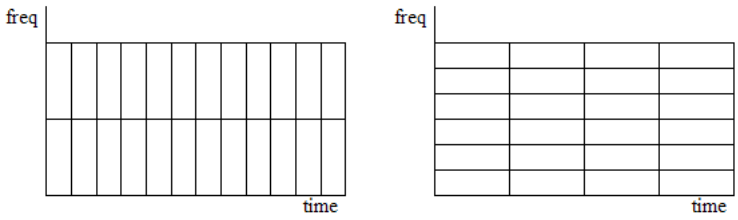


Figure 2.7: Trade off between the temporal and frequency resolution [56]

2.3.2 Wavelets

The wavelet has its roots with a Hungarian physicist Denis Gabor. Gabor was interested in decomposing a signal over elementary Time-Frequency "atoms". In 1946 he proposed replacing the analysis function of the Fourier transform with a complex exponential multiplied by a Gaussian window [17]. These Gabor atoms are used in the Gabor Transform to improve the Time-Frequency resolution of the Short Time Fourier Transform.

Thirty years later, the French geophysicist Jean Morlet developed a wavelet tool using the Gabor atom to analyze the vibrations from the earth's crust [23]. This method was further refined with the help of a French physicist Alex Grossmann [20]. It was with the collaboration of Grossmann and Morlet that the wavelet was formalized. A wavelet $\psi(t)$ is a wave that satisfies the following conditions

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

$$\int_{-\infty}^{+\infty} |\psi(t)|^2 dt = 1,$$

which implies that it is concentrated in time. Unlike complex exponentials which have

infinite energy for all time, wavelets have finite energy concentrated around a point. This property is advantageous for analyzing transient signals. Wavelets are used in many practical applications such as image compression, computer vision, signal processing and harmonic analysis.

There are many different types of wavelets, the ones that pertain to wavelet analysis can be divided into real and analytic. A real wavelet exists in the real-time domain, and must satisfy the admissibility condition

$$0 < C_\psi = \int_0^{+\infty} \frac{|\hat{\psi}(\omega)|^2}{\omega} d\omega < +\infty, \tag{2.9}$$

where: $\hat{\psi}(\omega)$ is the Fourier transform of the wavelet function, and C_ψ is the admissibility constant. The admissibility condition ensures that the wavelet is complete and observes energy conservation. It also implies that $\hat{\psi}(\omega)$ is differentiable, and $\hat{\psi}(0) = 0$. Real wavelets are used to detect transient frequencies in a signal and analyzing fractals. The Mexican Hat wavelet is a class of real wavelets which is defined as

$$\psi(t) = \frac{2}{\sqrt{3}} \pi^{-1/4} (t^2 - 1) e^{\frac{-t^2}{2}}.$$

A wavelet is analytic if it exists in both the real and the complex domains where

$$\hat{\psi}(\omega) = 0 \quad \text{if } \omega < 0.$$

They are used to analyze the time evolution of frequency tones by separating the amplitude and phase information of a signal. The Gabor wavelet is a class of analytic wavelet functions which is further explored in the following section.

The wavelet functions can be compressed and expanded to generate daughter wavelets with a scale parameter s, which can be any positive and real number. When the scale parameter increases so does the expansion of the wavelet. The addition of the scaling factor must be normalized by

$$\psi_s(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t}{s}\right) \quad s > 0 \tag{2.10}$$

to ensure that no additional energy is introduced to the function [29].

Morlet Wavelet

The Gabor wavelet is a class of functions that is defined by the multiplication of a complex exponential and a Gaussian window. In the literature, the complex representation of the wavelet function is known as the Gabor wavelet, while the real representation is known as the Morlet wavelet [26]. The Morlet wavelet is defined as

$$\psi_\phi(t) = \pi^{-\frac{1}{4}} C_\phi e^{-\frac{t^2}{2}}(e^{i\phi t} - \kappa_\phi), \tag{2.11}$$

where the admissibility constant C_ϕ and the correction factor κ_ϕ are given by [4]

$$C_\phi = (1 + e^{-\phi^2} - 2e^{-\frac{3\phi^2}{4}})^{-\frac{1}{2}}, \tag{2.12}$$

$$\kappa_\phi = e^{-\frac{\phi^2}{2}}. \tag{2.13}$$

The ϕ parameter is the central frequency of the Morlet wavelet measured in radians per second. The larger the parameter the higher the frequency (see Figure 2.9). The

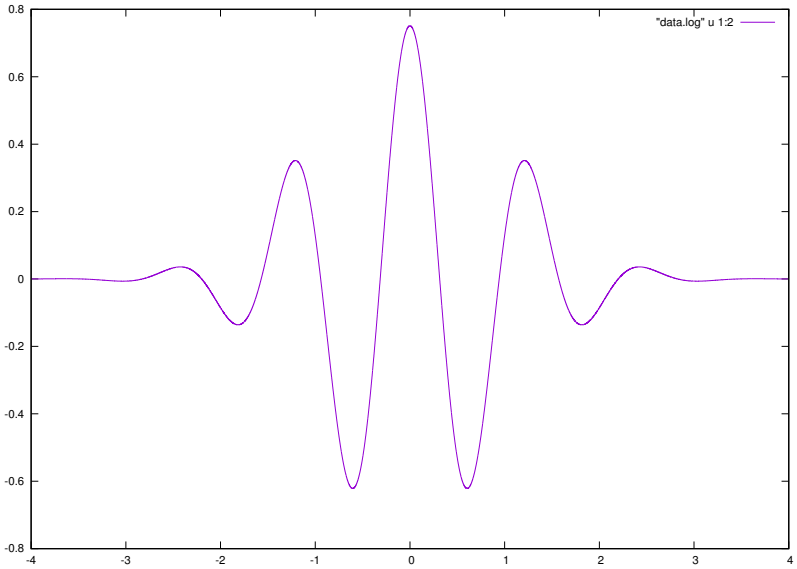


Figure 2.8: The Morlet wavelet in the time domain

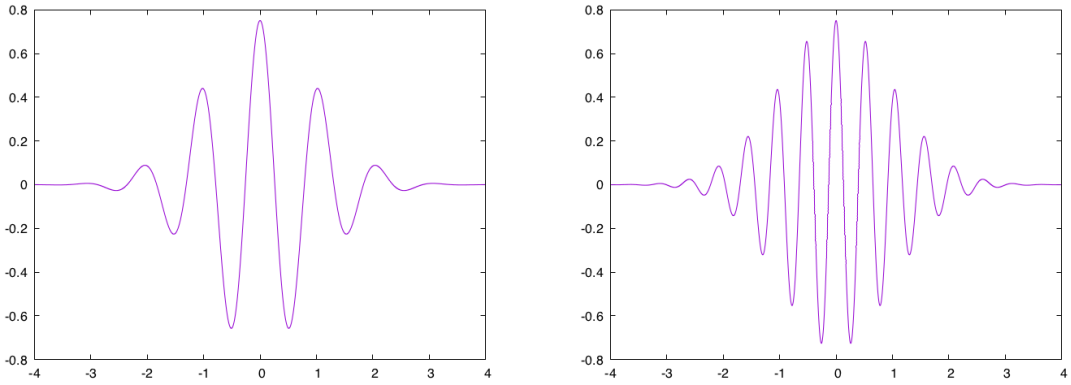
Morlet wavelet satisfies the conditions of a wavelet because it has finite energy, a mean of zero, and a square norm of one (see Figure 2.8). The Fourier transform of the Morlet wavelet is

$$\hat{\psi}_\phi(\omega) = C_\phi \pi^{-\frac{1}{4}} (e^{-\frac{1}{2}(\phi-\omega)^2} - \kappa_\phi e^{-\frac{1}{2}\omega^2}), \tag{2.14}$$

where the factors C_ϕ and κ_ϕ are the same as equations 2.12 and 2.13 respectively. The Morlet wavelet in the frequency domain is visualized in Figure 2.10.

When scaled the Morlet wavelet in the time domain can be normalized by the relation outlined in equation 2.10. In the frequency domain, the normalization is the inverse of the temporal domain [29], or

$$\hat{\psi}_{\phi,s}(\omega) = \sqrt{s}\psi_\phi(\omega). \tag{2.15}$$



(a) $\phi = 6.0$ (b) $\phi = 12.0$

Figure 2.9: The Morlet wavelet at different central frequencies

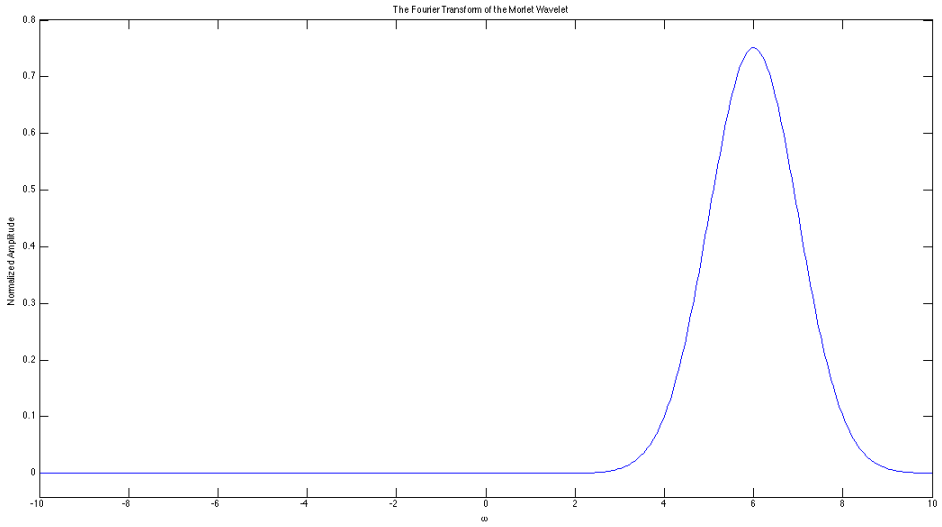
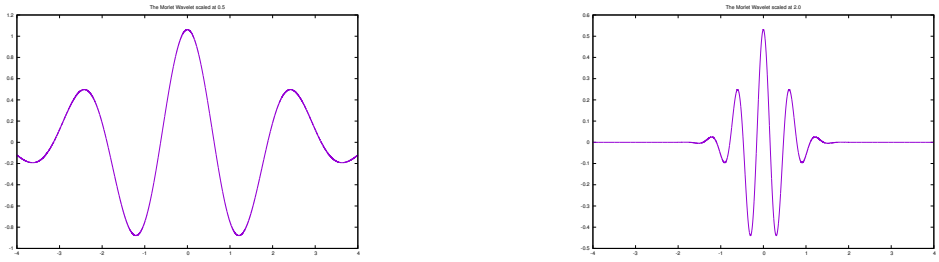


Figure 2.10: The Morlet Wavelet in Frequency Space



(a) s = 2.0 (b) s = 0.5

Figure 2.11: The Morlet Wavelet at different scales

Figure 2.11 shows the effect of scaling the wavelet.

In order for the Morlet wavelet to be computed automatically, equation 2.11 must be discretely sampled as

$$\psi_{\phi,s}[n] = \frac{1}{\sqrt{s}}\pi^{-\frac{1}{4}}C_\phi e^{\frac{(n\delta t)^2}{2s}}(e^{\frac{i\phi n\delta t}{s}} - \kappa_\phi) \quad n = 0, 1, 2, \ldots N-1 \tag{2.16}$$

where δt is the time between each sample and is defined as the inverse of the sampling frequency $\delta t = F_s^{-1}$. The wavelet in the frequency domain is sampled in the discrete time domain by the following relation

$$\hat{\psi}_{\phi,s}[n] = \sqrt{s}C_\phi\pi^{-\frac{1}{4}}\left(e^{-\frac{1}{2}(\phi-(sn\delta\omega))^2} - \kappa_\phi e^{-\frac{1}{2}(sn\delta\omega)^2}\right) \quad n = 0, 1, 2, \ldots N \tag{2.17}$$

where $\delta = 2\pi F_s/N$ measured in radians per second. The advantage of using Morlet wavelets is that the scales can be correlated with their corresponding continuous frequencies by the following relationship

$$f = \frac{\phi}{2\pi s}. \tag{2.18}$$

2.3.3 Continuous Wavelet Transform

The Fourier transform uses the complex exponential as its analysis function to decompose a signal. The complex exponential contains energy at all times so it is ineffective at localizing a signal in the temporal domain. The Continuous Wavelet Transform, or CWT, instead uses wavelets as its analyzing function. Since wavelets are localized in time and finite in energy, they are ideal for decomposing a signal in both time and frequency. The CWT decomposes a signal over elementary Time-Frequency atoms or wavelets, which can be used to construct a Time-Frequency representation of a signal [36]. It accomplishes this by performing multiple convolutions of the signal and wavelet function at different scales. The convolution of two functions $x(t)$ and $\psi(t)$ is defined as

$$(\psi * x)(t) \equiv \int_{-\infty}^{+\infty} x(t)\psi(t-\tau)d\tau,$$

and the discrete convolution of $x[n]$ and $\psi[n]$ is

$$(\psi * x)[n] = \sum_{k=0}^{N-1} x[k]\psi[n-k].$$

The CWT is the convolution of a signal $x(t)$ and the complex conjugate of different scaled daughter wavelets $\overline{\psi}(t/s)$ [26, 36]

$$\mathcal{W}_{\phi}(\tau, s) = \frac{1}{\sqrt{s}} \int_{-\infty}^{+\infty} x(t)\overline{\psi}\left(\frac{t-\tau}{s}\right) dt.$$

where: τ is the time delay measured in seconds and s is a real and positive scaling factor. The CWT can be approximated in the discrete time domain as multiple

convolutions of $x[n]$ and different variations of $\overline{\psi}[n]$ given as

$$\mathcal{W}[k,s] = \frac{1}{\sqrt{s}} \sum_{k=0}^{N-1} x[k]\overline{\psi}\left[\frac{n-k}{s}\right]. \tag{2.19}$$

The original signal $x(t)$ can be reconstructed by performing the inverse CWT which is given as

$$x(t) = \frac{1}{C_\psi} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \mathcal{W}_\phi(\tau, s)\psi\left(\frac{t-\tau}{s}\right) \frac{dsd\tau}{s^2}$$

where C_ψ is the admissibility condition outlined by equation 2.9 [9]. Like the STFT, the CWT is normally visualized by its energy spectrum density which is given as

$$\mathcal{P}(\tau, \omega) = |\mathcal{W}(\tau, \omega)|^2.$$

In order for a wavelet to be used in the CWT, the wavelet must satisfy the condition

$$\hat{\psi}(0) = \int_{-\infty}^{+\infty} \psi(t) = 0$$

which implies that it has a zero-mean. Another constraint is that the modulus of the wavelet function $|\hat{\psi}(\omega)|$ decreases to zero for $\omega \to \pm\infty$.

Unlike the Short Time Fourier Transform, the scaling of the daughter wavelets changes the wavelet's Time-Frequency resolution, making the Time-Frequency windows more dynamic. Since all of the wavelets have the same energy, large amplitude and short duration wavelets can be used to localize short high frequency information, while low amplitude and long duration wavelets can be used to localize the lower

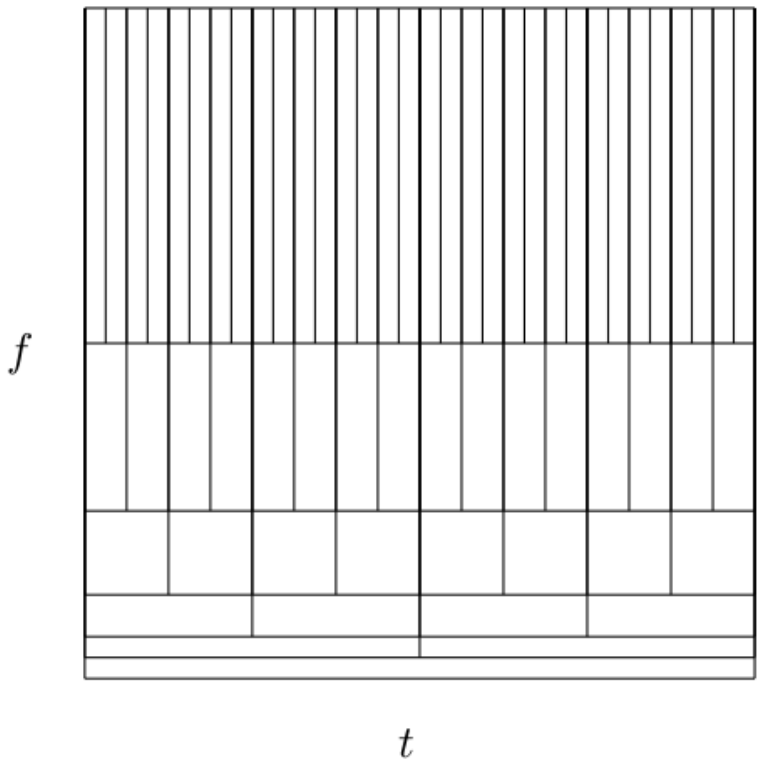


Figure 2.12: The changing Time-Frequency Windows of the CWT [7]

frequencies. Figure 2.12 shows the method of varying the window sizes to analyze both low and high frequency information. Flat windows at the bottom of the image show the window size for analyzing low frequency information, and long and narrow windows show the window size for analyzing high frequency information.

Chapter 3

Computation

In order for a signal processing method to be robust, it must be able to analyze a variety of signals automatically. While the previous chapter described the mathematical concepts of Time-Frequency analysis, these methods must be numerically approximated in order to be useful. This section outlines the algorithms to generate the Time-Frequency decompositions automatically. Section 3.1 describes how the STFT can be algorithmically computed. Section 3.2 describes how one can compute the CWT, which includes choosing the right wavelet and scaling functions. Finally, Section 3.3 describes the use of padding to improve runtime speed and transform accuracy.

3.1 Short Time Fourier Transform

The computation of the Short Time Fourier Transform was performed with the Rectangular Windowing Function. The Rectangular Window Function maintains the signal within a window, and replaces everything outside of it with zeros. If a discrete

signal $x[n]$ is of size N, and the rectangular window is of size A, then the i^{th} window would be

$$h_i[n] = \begin{cases} x[n] & i < n < iA \\ 0 & else \end{cases} \tag{3.1}$$

It is assumed that $N > A$ and i is any positive integer such that $i \leq N/A$. The window function was implemented by dividing the signal into smaller equally sized segments, and then computing the DFT for each window. The algorithm for performing the STFT with a rectangular windowing function is

1. Divide the signal $x[n]$ into smaller windows
2. For each window:
   (a) Compute the DFT of the window
   (b) Compute the magnitude of the DFT
   (c) Store the result

The data flow for the STFT is shown in Figure 3.1. In the data flow diagram it can be seen that the signal of size N is divided into J windows of size a samples, where $J = \lceil N/a \rceil$. The window size is left as an adjustable variable so the optimal window for the signal can be determined by the user. These smaller windows are then copied into a complex data array of size J and converted to the frequency domain using the DFT. The energy spectrum density is acquired by taking the magnitude of the array values. This is stored in a larger contiguous block of memory of size nJ which contains the energy spectrum densities for all previous and subsequent windows.

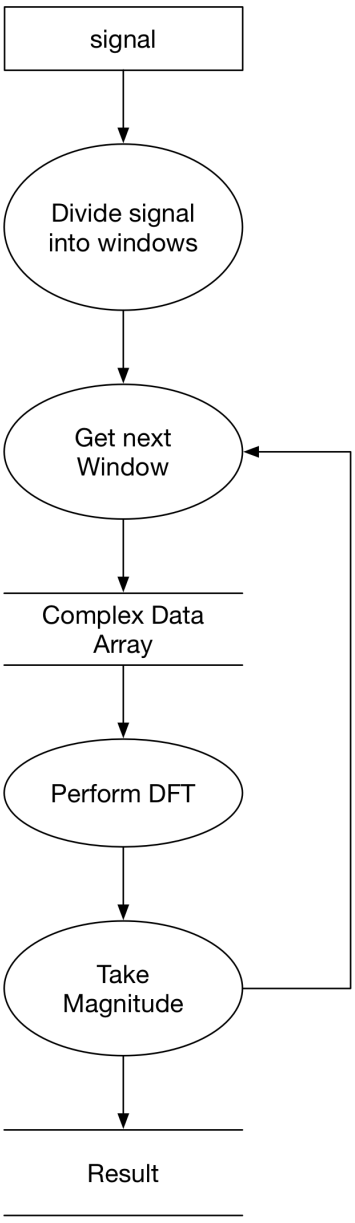


Figure 3.1: Data flow for the STFT

3.2 Continuous Wavelet Transform

Since the CWT can be used in a variety of applications, several decisions must be made to optimize it for Time-Frequency analysis. In this section we will discuss the choice of analyzing wavelet, and scaling function.

3.2.1 Choice of Wavelet

Choosing the right wavelet can improve computation speed as well as the effectiveness of the CWT. The vanishing moments, support size, regularity and selectivity in frequency are all considered when choosing a class of wavelet functions. A wavelet has p vanishing moments when

$$\int_{-\infty}^{+\infty} t^k \psi(t) dt = 0 \quad \text{for } 0 \leq k < p. \tag{3.2}$$

The wavelet's vanishing moment means that the wavelet $\psi(t)$ is orthogonal to any polynomial of degree $p-1$. High vanishing moments are ineffective for low frequencies and low vanishing moments are ineffective for high frequencies. Wavelets with high vanishing moments have larger supports which means they require greater computational cost. One would use higher vanishing moments for numerical analysis, and low vanishing moments for data compression. The regularity of the wavelet, which is the number of continuous derivatives a wavelet has, must also be considered. A wavelet's regularity has more effect on compression and encoding, so it is not a significant factor for choosing a wavelet for signal analysis [36].

The wavelet function for the Time-Frequency decomposition must have high vanishing moments, and must also be highly selective in the frequency domain. The

Morlet wavelet was chosen because it satisfied both of these constraints. In fact, the Morlet wavelet is known to be Time-Frequency optimal [58]. To ensure that the Morlet wavelet maintains its selectivity in the frequency domain, the central frequency parameter ϕ was set to be 6.0. The central frequency parameter increases the temporal resolution at the cost of frequency. Normally, this parameter is set between 5.0 and 6.0 to avoid floating point errors [29].

Time-Frequency Accuracy

A number of sources such as [14, 38, 57] represent the Morlet wavelet as

$$\hat{\psi}(\omega) = \pi^{-1/4} H(\omega) e^{-(s\omega-\phi)^2/2}. \tag{3.3}$$

Where $H(\omega)$ is a Heaviside step function given as

$$H(\omega) = \begin{cases} 1 & \omega > 0 \\ 0 & \omega < 0 \end{cases}$$

This definition discards the κ_ϕ term which is used to satisfy the admissibility condition. This is sometimes done in practical applications as the value is seen to be negligible in floating point operations [4]. However, this introduces numerical error which has to be corrected with a "Fourier Wavelength Factor". While the Morlet wavelet defined in Section 2.3.2 requires more computation, it was used over equation 3.3 for its improved Time-Frequency accuracy.

3.2.2 Choice of Scaling Function

The fourth A key from the left of the piano or A_4 has a dominant frequency of 440 Hz. A_3 — which is one octave below — has a frequency of 220 Hz, and A_5 has a frequency of 880 Hz. Sound does not scale linearly but rather it is exponential. C_5 is not twice the frequency as C_3 but four times its frequency. A piano has 12 keys in every octave, and its frequencies can be mathematically represented as

$$f(n) = 440\left(2^{\frac{1}{12}(n-49)}\right),$$

where: $f(n)$ is measured in Hertz, and A_4 is the 49^{th} key from the left [8]. Since A_4's dominant frequency is an integer, it is used as a basis for scaling the other frequencies.

Signals — such as sound and music — scale exponentially, so the wavelets that analyze them should scale in the same manner. The dyadic scaling function, defined by equation 3.4, was used because it scales by powers of two just like the frequencies on a piano. The scaling function is also an orthonormal basis which removes all redundancy when decomposing a signal in the Time-Frequency domain.

The dyadic scale as outlined by Mallet and Zhong was used to compute the Morlet wavelet at different scales [37]. The scales are powers of two of the smallest scale s_0. The smallest scale s_0 must be set such that the CWT can cover the entire Time-Frequency domain. This condition can be satisfied by setting s_0 to be sufficiently small [36], for convenience $s_0 = \delta t$. The largest scale needed to compute the lowest frequency is represented as J, which is computed as $J = \log_2 N$, where J is any positive integer. The dyadic scaling function is defined as

$$s_j = s_0 2^j \quad j = 0, 1, \ldots J. \tag{3.4}$$

To improve the frequency resolution of the wavelet transform, sub scales are added in between. These sub scales, known as "sub-octaves" are denoted by a δj. The dyadic scale then becomes

$$s_j = s_0 2^{j\delta j} \quad j = 0, 1, \ldots J. \tag{3.5}$$

Dyadic Scaling Function Error

The dyadic scaling function means that the error interval is also dyadic. If the scale is determined using equation 4.1, then the frequency is accurate to within

$$f_{error}(s) = \frac{\phi}{2\pi s_0 2^{s+\delta j}} - \frac{\phi}{2\pi s_0 2^{s-\delta j}} \tag{3.6}$$

where ϕ is the central frequency of the Morlet wavelet. The dyadic scales have smaller windows in the lower frequencies, and larger windows in the higher frequencies, so the error in the higher frequencies are larger. The resolution in the higher frequencies can be increased by changing the sub-octave parameter δj.

3.2.3 Algorithm

The convolutions of the CWT are computationally expensive, and can be computed in the frequency domain instead. This is done by using the convolution theorem, which states that a multiplication in one domain is the convolution in the other and vice versa [15]. The convolution of the wavelet and data functions are performed by first multiplying them in the frequency domain, and then transforming it back into the time domain by taking the IDFT. The algorithm to perform the CWT is

1. Compute the DFT of the signal

2. For every scale:

   (a) Compute the Fourier transform of the Morlet wavelet

   (b) Multiply the wavelet and signal in the frequency domain

   (c) Compute the IDFT of the multiplication

   (d) Compute the magnitude and normalize with respect to the scale

Figure 3.2 shows the data flow of the CWT algorithm. The signal is first copied into a block of contiguous memory of size n. It is then copied into a two dimensional FFT array with real and complex values. Usually, the padding would be added around the signal; but since the DFT is cyclic, all of the padding can be added to the end. The DFT is then computed on the FFT array. The Fourier transform of the Morlet wavelet at the specified scale is computed and stored in a complex two-dimensional array. The Fourier transform of the Morlet wavelet and the signal are then multiplied in the frequency domain and stored in a resulting data array. The IDFT of this multiplication is then computed to perform the convolution. The magnitude of the convolution is taken discarding the padding to obtain the energy spectrum density. This is then stored in a larger contiguous memory block that contains all of the energy spectrum densities for every scale.

3.3 Padding

Edge effects can sometimes be introduced when performing the FFT of a signal. Since the FFT operates on a finite number of samples, when the DFT and IDFT are performed, the signal is transformed from the frequency domain into the time

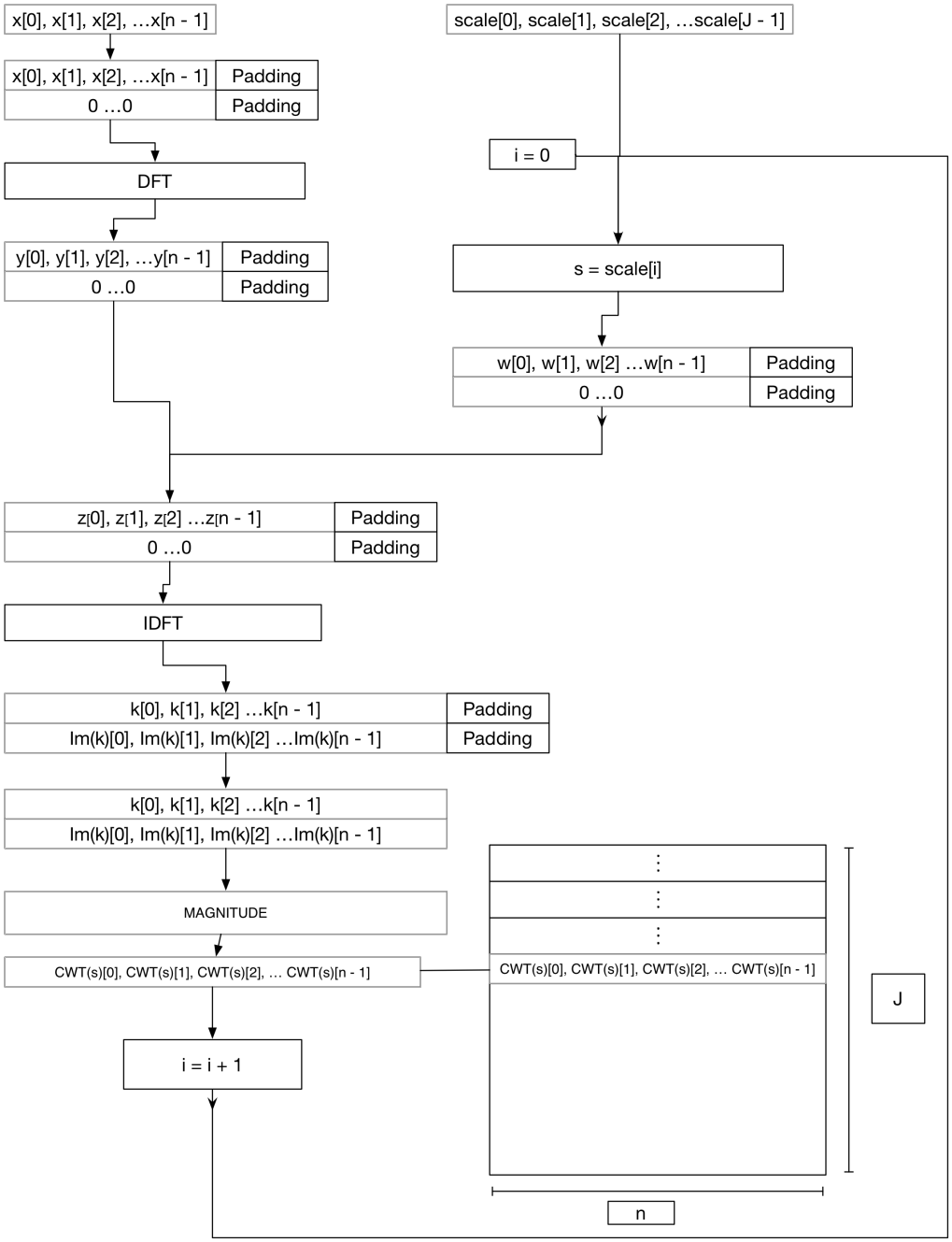


Figure 3.2: Data flow for the Wavelet Analysis

domain as a periodic signal. This introduces errors in the first and last samples of the signal known as edge effects [48]. This issue can be mitigated by padding the signal to improve the accuracy of the FFT. This section discusses the different methods of padding the signal before transforming it into the frequency domain.

3.3.1 Zero Padding

The most common method of padding is to introduce zeros to the signal. Before computing the Fourier transform, the number of samples are increased to the closest power of two. The signal is then shifted to the center of the array and zeros are added on either side. This method allows the use of the Cooley-Tukey FFT algorithm [22, 50]. To zero pad a signal the size of the signal array is increased by a factor of

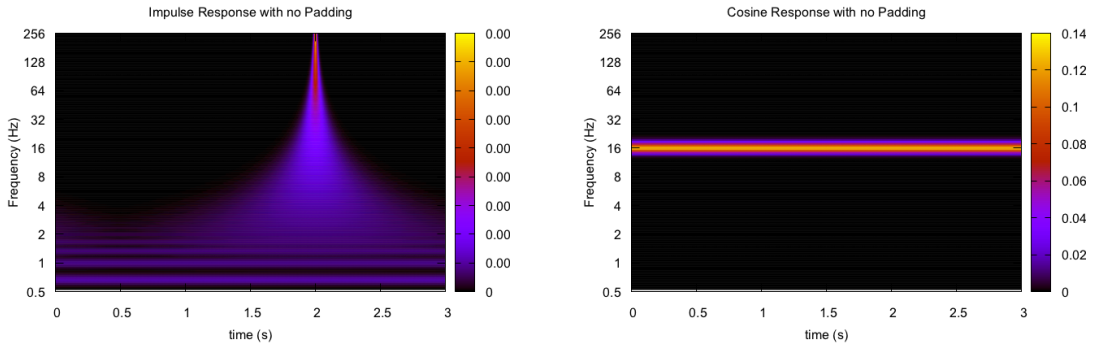
$$N_{pad} = 2^{\lceil \log_2(N+1) \rceil}. \tag{3.7}$$

3.3.2 Ramp Padding

This method doubles the size of the signal array from N to $2N$. It ramps down the first half of the original signal at the end of the array and ramps up the second half of the signal before the beginning. If the original signal $x[n]$ spans from 0 to A then the ramp padded signal $r[n]$ is

$$r[n] = \begin{cases} \alpha(n)x[N/2+n] & 0 < n < N/2 \\ x[n-N/2] & N/2 < n < 3N/2 \\ (1-\alpha(n))x[n-3N/2] & 3N/2 < n < 2N \end{cases} \tag{3.8}$$

The ramping factor α is a linear ramp from zero to one or



(a) An Impulse Response with no padding (b) The Cosine Response with no padding

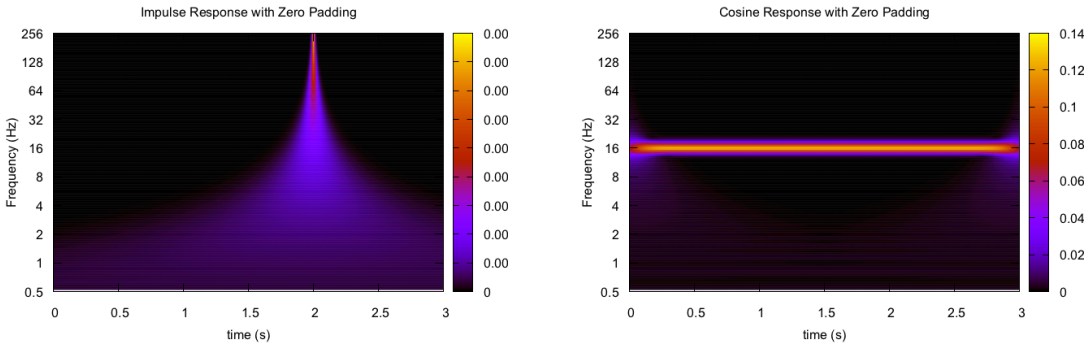
Figure 3.3: Figures of the accuracy of without padding

$$\alpha(n) = \left(\frac{N}{2}\right)^{-1} n \quad 0 < n < N/2$$

3.3.3 When to Use Padding

It is common practice to pad the signal array when computing its DFT, as this can increase the speed and accuracy of the computation. The CWT can be computed with or without padding, but this choice can change the accuracy of the decomposition.

One would notice edge effects if a periodic signal is padded with zeros. The CWT of a periodic and an aperiodic signal was computed, the functions were sampled at 2048 Hz for 3.0 s. The periodic signal is represented by a constant cosine, and the aperiodic signal is represented by an impulse at 2.0 s. Figure 3.3 shows the effect of not padding a signal. Bands can be seen on the lower frequencies of the impulse response, while the periodic cosine function does contain artifacts. The artifacts are caused from not padding the aperiodic signal. The inverse relation can be seen when computing the CWT with zero-padding.



(a) An Impulse Response with padding (b) The Cosine Response with padding

Figure 3.4: Figures of the accuracy of with padding

Figure 3.4 shows the same functions computed with zero-padding. Zero padding does not introduce artifacts for aperiodic signals (Figure 3.4a), but introduces artifacts near the edges of the lower frequencies for periodic signals (Figure 3.4b). Padding a periodic signal, or not padding an aperiodic signal will introduce additional noise and inaccuracies in the analysis process, therefore the type of signal must be considered before analyzing it with padding.

Chapter 4

Implementation

To make the framework useful, the computation of the Time-Frequency methods were implemented in the C++ programming language. This section outlines the tools and methods that were used to perform the Time-Frequency computation. In addition, speed up and accuracy methods are also discussed.

4.1 Choice of Programming Language

The algorithms outlined in Chapter 3 can easily be implemented in a numerical computing environment such as MATLAB or Octave. While this process may be easy, it sacrifices speed and efficiency. The Time-Frequency methods can be made faster through low level memory manipulation. This is the reason that the algorithms were implemented in the C and C++ programming languages.

The C++ language is often used in industrial applications which would allow the Time-Frequency framework to be easily integrated. Finally, the C++ language has many libraries and tools the can be used, such as FFTW and PNGWriter.

4.2 Fast Fourier Transform

The open source library FFTW or Fastest Fourier Transform in the West was used to compute the forward and inverse FFTs in the algorithms because of its speed and accuracy [16]. The choice of using FFTW allows several computational optimizations to be used such as reducing the need to compute FFT coefficients repeatedly.

The FFTW library requires two levels of computation to work. The first step is to plan the method of performing the FFT. The second step is executing the plan, which can be performed multiple times. The planning process consists of choosing the most efficient algorithm for the particular computing architecture, and pre-computing the coefficients that are required by the FFT. Separating the FFT into these steps, allows these coefficients to be computed once and reused. The same coefficients can be used for different data arrays, as long as the size of the data array does not change.

The computation of the Time-Frequency analysis can use this method to improve the speed. Once the complex data arrays are allocated, the FFTW plans for the forward and inverse DFTs are computed. The forward plan is used once to take the FFT of the signal, and the inverse plan is used repeatedly to convert the convolutions back into the time domain. This step does not require us to compute the FFTW coefficients every time which saves significant computational resources.

Another interesting tool available when using FFTW is the ability to store wisdom, this is where FFTW tests multiple FFT algorithms and determines the fastest algorithm for that specific case. The stored wisdom can be used to compute subsequent Fourier transforms more efficiently. CWTs require many Fourier transforms, so the wisdom feature was used to improve the speed. Before the CWT is computed, the `Generate_FFTW_Wisdom()` function can be called to compute the optimal FFT

algorithm. This wisdom is then used to compute the subsequent FFTs.

One final note about computing the FFTs is the choice of using FFTW's complex one-dimensional method, or `fftw_plan_dft_1d()`. The library offers the faster option of performing a one-dimensional DFT of real data. While one can improve the speed and memory usage by a factor of two using this method, it was not suitable for computing the DFTs for Time-Frequency analysis. Since the Time-Frequency decomposition has energy distributed over both the real and complex domain, it is vitally important to compute the DFTs with the complex one-dimensional method instead.

4.3 Scale Limits

The frequency of the CWT can be band limited such that only the relevant information is analyzed to decrease the amount of computation. This method would be useful when used with EEGs where one would limit the analysis of the signal to the maximum and minimum neuron firing rates of 70 Hz, and 1 Hz respectively. Once there is an upper and lower band limit, the scales and sub-scales are computed using equation 3.5. The corresponding scale given a frequency is determined by the following relationship

$$s = \left\lceil \frac{\log_2\left(\frac{\phi}{2\pi s_0 f}\right)}{\delta j} \right\rceil . \tag{4.1}$$

4.4 Convolution

To reduce the computation and memory cost of storing conjugate complex values, FFT algorithms store the Fourier coefficients as

$$z_k = \overline{z}_{N-k}.$$

The positive frequencies are stored in the first half of the output in order, and the negative frequencies are stored in the second half of the array in reverse order [49]. The frequency array of an FFT algorithm would be ordered as

$$x[0], x[1], x[2], ...x[N/2], \overline{x}[(N+1)/2-1], ...\overline{x}[2], \overline{x}[1].$$

The convolutions of the Time-Frequency decompositions are computed with complex arrays. So they must be multiplied in the frequency domain with the negative frequencies in mind. The method to multiply two data arrays in the frequency domain is

1. Multiply $\hat{x}[0]$ and $\hat{\Psi}[0]$.
2. for $i = 1..N/2-1$
   (a) Multiply $\hat{x}[i]$ and $\hat{\Psi}[i]$
   (b) Multiply $\hat{x}[i]$ and $\hat{\Psi}[N-i]$

The choice to use analytic wavelets allows this process to be further optimized. All negative frequencies of an analytic wavelet are zero. Computing the wavelet in the negative frequencies, and multiplying them with the data is unnecessary. These values

can be set to zero to decrease computation time without any impact on accuracy.

4.5 Plotting

The Time-Frequency decompositions generate data in the time and frequency dimensions. In order to map these values onto an image, careful usage of logarithmic mapping and colour schemes must be used. The results must be plotted linearly in the time domain, with a $\log_2$ scale in the frequency domain, and a $\log_{10}$ scale in the normalized amplitude domain. The amplitudes of the energy spectrum densities must first be converted into colours to represent their relative normalized amplitudes with respect to each other.

The plotting of the Time-Frequency decompositions is accomplished with the interactive plotting interface `gnuplot` [25], it can also be saved as a PNG using the `PNGWriter` library. PNGWriter allows the plotting and reading of standard color spaces such as RGB and HSV. The library has additional tools to plot basic shapes and curves like squares, diamonds and Bezier curves. To plot the energy spectrum density of the Time-Frequency analysis, the values from the array are plotted pixel by pixel. The size of the image is first computed as a function of the size of the energy spectrum density array. The image width was calculated using an O_x and s parameter

$$w = 2O_x + (N/s).$$

where: N is the number of samples in the time domain, O_x is the number of white pixels that will be on either side of the image, and s is the stride parameter.

The stride parameter is used to ensure that the image does not stretch horizontally. If a signal is long in duration or has a high sampling rate then the energy spectrum density of that image would appear squashed as the length of the temporal domain is much longer than the frequency domain. The stride parameter determines the number of samples to skip in the time domain to make the image more proportional.

The image height was calculated using an O_y and l parameter

$$h = 2O_y + (lJ)$$

where: J is the number of scales the Time-Frequency analysis was conducted on, O_y is the number of white pixels that are placed above and below the energy spectrum density graph, and l is the line size parameter which is used to increase the visibility of the frequency resolution.

There is more temporal data than frequency data in the Time-Frequency decomposition. If plotted as-is, the image would be too long in the time domain, and very short in the frequency domain. This would make it difficult for a user to distinguish the frequency components of a signal. The l parameter repeats the information of each frequency scale l times to increase the vertical length without having to increase the number of sub-octaves the wavelet transform must compute. Increasing the l parameter adds vertical height to the image which allows greater visual fidelity in the frequency domain. The energy spectrum density is plotted after the image size is established.

The energy spectrum density is computed by first taking the logarithm of all of the values. This is done using the `log10()` function from the `<math.h>` standard library [46]. The logarithmic values are represented as a colour from blue to red,

where blue is the lowest value and red is the highest value in the energy spectrum density graph. PNGWriter plots a pixel given the amount of red, blue and green that pixel needs, which are values between zero and one. The colour of a pixel, represented as $C(r, g, b)$, is determined by the following relationship

$$C(r) = \begin{cases} 4(v - min)/range & v > \frac{3}{4}range \\ 0 & else \end{cases}$$

$$C(g) = \begin{cases} 4(v - min)/range & v < \frac{1}{4}range \\ 1 + 4(min + 0.75range - v)/range & v > \frac{3}{4}range \\ 0 & else \end{cases}$$

$$C(b) = \begin{cases} 1 + 4(min + 0.75range - v)/range & \frac{1}{4}range < v < \frac{1}{2}range \\ 0 & else \end{cases}$$

where: v is the value to be converted, and $range$ is the difference between the maximum and minimum values in the array. Figure 4.1 shows the colour bar ramping from 0.0 to 1.0. The image is generated by converting the values from the energy spectrum density array to their corresponding color and then placing these colors onto the PNG file pixel by pixel. Since PNGWriter references pixels from the top left corner of the image, the energy spectrum density graph was plotted backwards, where the lowest frequencies are plotted first and then the higher frequencies.

Markers and labels are then placed at regular intervals to mark the frequency and time points on the axis. The interval marks on the axes were generated using PNGWriter's `filledsquare()` function. To ensure that the interval marks were

accurate, the squares were drawn with the referential origin at the midpoint (Figure 4.2).

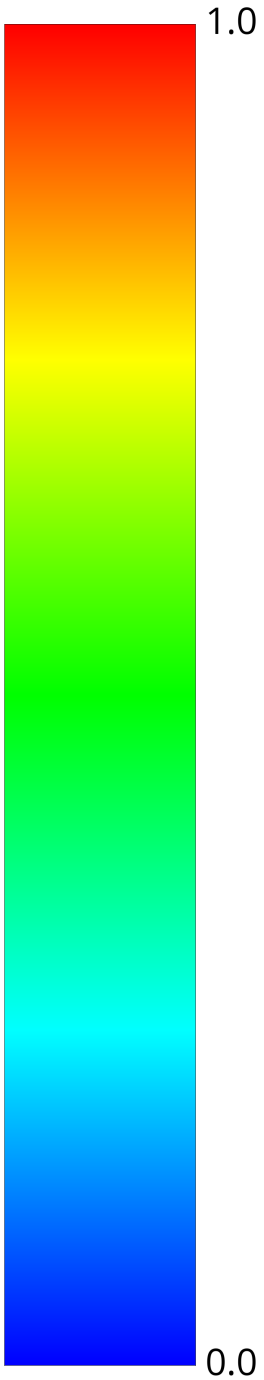
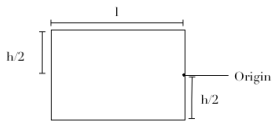
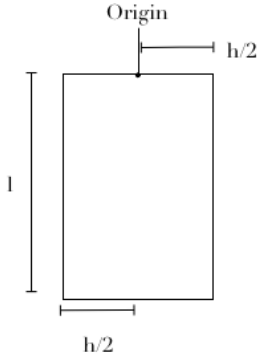


Figure 4.1: The Colour Ramp from 0.0 to 1.0



(a) Frequency



(b) Time

Figure 4.2: Interval Marker Geometry

Chapter 5

Validation

The implementation of the Time-Frequency framework was validated by conducting a series of tests with expected results. FFTW Version 3.3.6 was used with the FFTW plan flag set to `FFTW_ESTIMATE`, which provided a reasonable execution time to validate the framework. The specifications of the test computer is outlined in Table 5.1

CPU	2.7 GHz Intel Core i7
Memory	16.0 GB 1600 MHz DDR3
GPU	Intel HD Graphics 3000 512 Mb
Storage	Western Digital 500 GB Solid State SATA Drive
Operating System	OSX 10.11.6
Compiler	GCC 5.4.0
Optimization	`-O2`

Table 5.1: Test Computer Properties

5.1 Sampling Rates and Signal Length

The results of the decompositions do not vary with the sampling rate or signal length. In order to test this, a cosine and Kronecker impulse function were sampled at rates between 100 to 20 000 Hz. Separate tests simulated a signal varying between 1.0 s to 15.0 s. As expected, the wavelet transforms did not shift in the frequency or time domains.

5.2 Sinusoidal

The CWT of a sinusoidal at a known frequency and amplitude was computed to test if the same amplitude and frequency were observed in the analysis process. The CWT of the signal

$$x(t) = Acos(\omega_0 t), \tag{5.1}$$

was computed. The CWT of equation 5.1 resembles the FFT as the maximums are observed at ω_0 with a magnitude of A. The CWT, however, provides both temporal and frequency resolution, so the maximum at ω_0 is observed for the entirety of the signal. This test was conducted to confirm that the CWT does not shift in the frequency and magnitude domain. The system was also introduced with cosines of varying sample lengths and the results were unchanged.

5.3 Impulse Response

The CWT of an impulse function

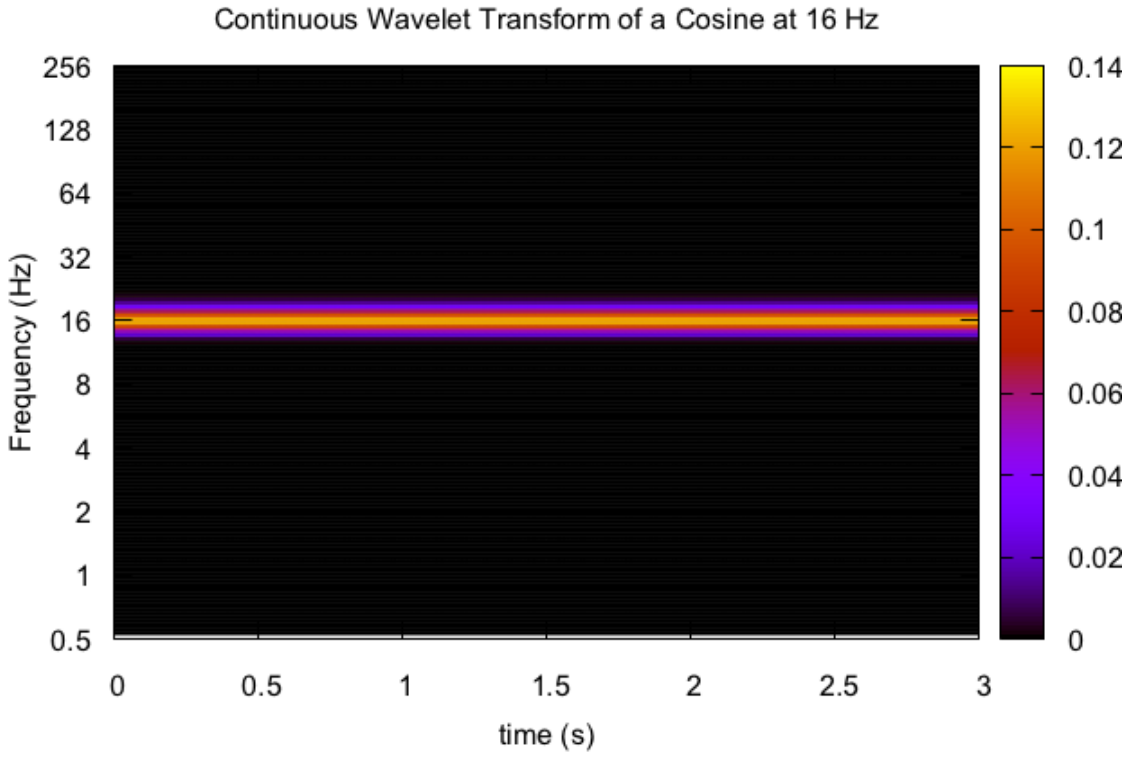


Figure 5.1: The CWT of a cosine function.

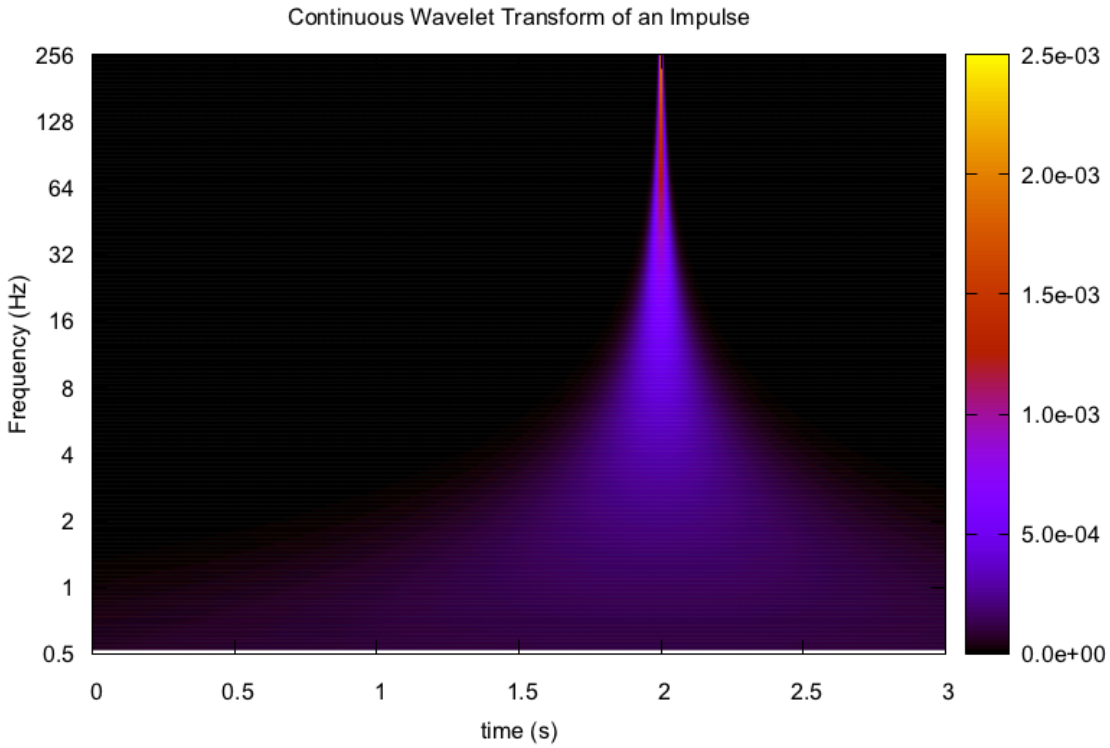


Figure 5.2: The CWT of an impulse function.

$$x[n] = \delta[n - \tau],$$

was computed to ensure that it was accurate in the time domain. The CWT had a global maximum at $t = \tau$, which means the computation does not shift the signal temporally.

5.3.1 Phased Cosine Response

The CWT of a sinusoidal with a phase offset was computed, the magnitude and frequency domains should not be affected by this offset. The Time-Frequency decomposition of the signal

$$x(t) = Acos\left(\omega_0 t + \frac{\pi}{4}\right) \tag{5.2}$$

was computed. The global maximum of the CWT of $x(t)$ was equal to A, and was observed at the frequency ω_0.

5.4 Multiple Frequencies

A piecewise sinusoidal was analyzed to test both the STFT and the CWT's ability to analyze the Time-Frequency information of a signal. The Time-Frequency analysis of

$$x(t) = \begin{cases} cos(\omega_0 t) & 1.5s \leq t \leq 2.0s \\ cos(\omega_1 t) & t < 1.5s \quad t > 2.0s \end{cases} \tag{5.3}$$

where: ω_0 and ω_1 was set to two different frequencies. Both Time-Frequency analysis methods were able to isolate and identify the correct frequencies for the appropriate time durations.

5.5 Numerical Experiments

The computed results were compared to analytic CWT functions to identify the amount of error. The absolute error is defined as [3, 21]

$$e_{abs} = |f(x)_{computed} - f(x)|. \tag{5.4}$$

This definition was used to compare the analytic and computed results of the Time-Frequency analysis methods. The CWT of an impulse and a cosine were computed analytically, and then compared with their CWT approximations to determine the amount of absolute error.

5.5.1 Impulse

The CWT of an impulse at 2.0 sec was analytically computed as [29]

$$\mathcal{W}(\tau, s) = C_\phi \pi^{-\frac{1}{4}} \frac{1}{\sqrt{s}} \left[e^{-\frac{1}{2}\left(\frac{2-\tau}{s}\right)^2} \left(e^{i\phi\left(\frac{2-\tau}{s}\right)} - \kappa_\phi \right) \right]. \tag{5.5}$$

When using zero padding the maximum absolute error is shown in Table 5.2

5.5.2 Cosine

The CWT of a cosine, such as equation 5.1, was computed analytically. The transform is determined to be [29]

$$\mathcal{W}(\tau, s) = C_\phi \frac{\sqrt{s}}{2} \pi^{-\frac{1}{4}} \left[e^{\frac{-1}{2}(\phi - s\omega_0)^2} + \kappa_\phi e^{\frac{-1}{2}(s\omega_0)^2} \right] e^{is\omega_0\tau}. \tag{5.6}$$

Function	Zero Padding	No Padding
Impulse	6.60×10^{-14}	4.07×10^{-5}
Cosine	4.59×10^{-2}	7.46×10^{-10}

Table 5.2: Absolute errors obtained from the numerical experiments

Transform	Impulse (s)	Cosine (s)
STFT	1.5×10^{-4}	1.5×10^{-4}
CWT	8.16×10^{-2}	8.27×10^{-2}

Table 5.3: Computation times for the Time-Frequency methods

The absolute error between the analytic and computed cosine response is shown in Table 5.2.

5.6 Execution Time

The framework was timed by running an impulse and cosine function 10 000 times and dividing the total compute time by 10 000. The signals were sampled at 2048 Hz for 3.0 s. The STFT was set to a window size of 500 samples over the entire frequency band. The Continuous Wavelet Transform was computed with a Morlet wavelet with a minimum frequency of 0.5 Hz, and a maximum frequency of 1024 Hz. The wavelet was scaled dyadically, with a sub-octave resolution of 0.03125. Table 5.3 shows the execution times for both the STFT and the CWT.

Chapter 6

Application

This chapter describes the two scenarios where Time-Frequency analysis was used. We first explore the current tools available in neuroscience for computing the Event Related Spectral Perturbation (ERSP). We then apply the CWT to improve the reliability of the ERSP computation. In section 6.2 Time-Frequency analysis is used to identify the mechanical parameters of an oscillating machine. Data was analyzed from research done within our lab group to identify a machine's eigen frequencies and their settling times.

6.1 EEG Analysis

In this section we provide begin with a background review of EEG analysis before describing the role of Time-Frequency analysis in EEG analysis. We first describe the use of EEGs, and then describe two methods of analyzing EEG signals, namely Event Related Potentials (ERPs), and Event Related Spectral Pertubation (ERSP). Then in section 6.1.4 we describe the method of using Time-Frequency analysis to

improve the analysis process of EEG signals when performing ERSP computations. The remainder of the section describes how this method is implemented within the framework.

6.1.1 Electroencephalography (EEG)

Electroencephalography (EEG) is the record of the electric discharges from neurons detected by electrodes on the surface of the scalp [43]. The method of recording electro-physiological responses comprises of electrodes, amplifiers and methods of storing the signal. EEGs can be recorded by invasive, or non invasive electrodes placed on the scalp (Figure 6.1) [30, 39]. EEG electrodes are a combination of a metal and their metallic salts as this provides reversible electron flow between the electrode and the skin [13]. The most common electrodes used for clinical purposes are made of silver/silver-chloride material [28].

The voltage fluctuations from EEGs are between 1 µV to 100 µV [30]. In most applications the signal is band limited to record between 0.5 to 70.0 Hz [60]. The signal from an EEG electrode is amplified and filtered to enhance these properties. The recordings are converted into a digital signal, and stored as a function of time in specialized data formats.

EEGs have been an important tool in understanding the function of the cerebral cortex. The frequencies at which neurons fire can help identify and diagnose what the brain is doing when exposed to a stimulus. Table 6.1 shows the different frequencies found in the brain and the general types of mental states associated with each frequency [30].

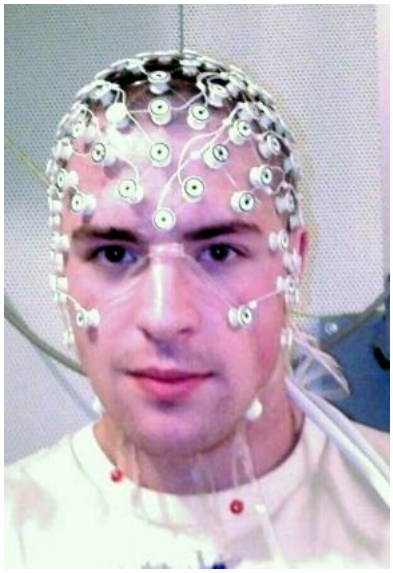


Figure 6.1: A patient wired with EEG electrodes [40]

Wave Type	Frequency (Hz)	Description
δ	0.5 - 4.0	Deep REM sleep
θ	4.0 - 7.0	Linked to memory formation
α	8.0 - 15	Daydreaming and relaxing
β	15 - 30	Normal waking consciousness
γ	25 - 100	Attentively focusing

Table 6.1: Brain Activity and their corresponding frequencies found in EEGs

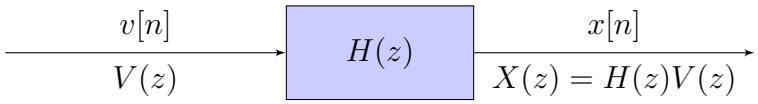


Figure 6.2: Linear Stochastic Model of EEG signals

Attempts have been made to mathematically model the electrical neuron discharges from the cerebral cortex. EEG signals have been mathematically modeled both deterministically and stochastically, but there is not much academic consensus on the correct modeling technique [51]. The most popular model that is used is the linear stochastic model, which models a signal as a linear system driven by Gaussian noise (Figure 6.2) [24].

A Gaussian noise function $V(z)$ which has a variance of σ_v^2 is linearly filtered by $H(z)$, which produces an output EEG signal $x[n]$. It is advantageous to understand the method of modeling EEGs as it provides a basis of developing useful algorithms for extracting the signal from the noise. There are further specifications for the linear stochastic model known as

- The Autoregressive Model
- The Autoregressive Moving Average Model
- The Autoregressive Model with an Impulse Input
- The Time-varying Autoregressive Model

In depth explorations of these methods are beyond the scope of this text, but the interested reader may wish to consult the textbook in [51].

6.1.2 Event Related Potential (ERP)

An Event Related Potential (ERP) is the measurement of the electro-physiological response by an organic body to an external stimulus. One can use ERPs to analyze a variety of biological signals such as Electromyography (EMG), Electrocardiography (ECG), and Electroencephalography (EEG). ERPs are conducted over many trials and then averaged to help reduce the random background noise. The source of the random noise can be from other bio-electric signals such as muscular, ocular and cardiac interference. It can also be sourced from electromagnetic interference such as the 60 Hz line noise found in North American electrical circuits. One method of removing the noise is through a process called ensemble averaging.

Ensemble averaging is based on the idea that the i^{th} trial of an ERP, given as $x_i(n)$, is the sum of the signal $s(n)$ and an uncorrelated signal of random noise $v_i(n)$. This is represented as

$$x_i(n) = s(n) + v_i(n).$$

The random noise $v(n)$ is assumed to be a zero-mean process

$$E[v(n)] = 0,$$

where $E[\cdot]$ is the mean function. The noise of the signal is represented by its correlation function $r_v(k)$ given as

$$r_v(k) = E[v(n)v(n-k)].$$

When the noise of the signal is random, its variance is fixed and identical throughout all of the trials

$$r_v(0) = E[v_i^2(n)] = \sigma_v^2, \quad i = 1, \ldots, N.$$

This property implies that the correlation function $r_v(k)$ decays to zero rapidly, so the noise can be seen as uncorrelated from trial to trial. This means that the signal can be added and averaged without having the noise accumulate. The correlation of two different trials then becomes

$$E[v_i(n)v_j(n-k)] = r_v(k)\delta(i-j),$$

where $\delta(i)$ is the Kronecker delta defined as

$$\delta[i] = \begin{cases} 1, & i = 0 \\ 0, & i \neq 0 \end{cases}. \tag{6.1}$$

The Event Related Potential $ERP(t)$ generated from ensemble averaging is defined as the sum over N trials of a stimulus response $F_k(t)$ shown as

$$ERP(t) = \frac{1}{N}\sum_{k=1}^{N} F_k(t).$$

By using ensemble averaging the noise is removed, and the signal remains where

$$E[ERP(N)] = s(n).$$

The variance of the ERPs is known to be inversely proportional to the number of trials N, which is given by

$$V[ERP(n)] = \frac{\sigma_v^2}{N}.$$

Ensemble averaging builds off of the linear stochastic model as it aims to subtract the random background noise from the signal. This is done by conducting the same experiment repeatedly and then averaging the responses.

When using ensemble averaging, one must assume that the linear stochastic model holds, which may not always be the case. For example, one of the assumptions is that the signals that are analyzed have a zero-mean. Unwanted noise may be introduce into the analysis process if the system is analyzed with non-zero means. They also assume that the signal is fixed and identical from trial to trial. This would be untrue if the subject's state is altered due to expectation, habituation, or environmental activity. ERPs can also fail when the signal $s(n)$ is correlated with the noise $v_i(n)$. This may occur when the subject is aware of the experiment and is anticipating a stimulus. Finally, the basis of ensemble averaging is that the noise has a predictable statistical distribution. This assumption would not hold if the trials have the subject blinking, which is known to introduce a lot of unwanted noise. Ensemble averaging can help extract information from a signal, yet there are limitations on how effective it can be in studying the cerebral cortex.

6.1.3 Event Related Spectral Perturbation

One can analyze the amplitude and phase of the Event Related Potentials, but these methods have limits in their ability to fully decompose the signal's information. EEG signals have non-stationary and transient properties, which necessitates temporal and frequency analysis. The transient nature of EEG signals are not suited for Fourier

analysis as the sines and cosines are not localized in time. Event Related Spectral Perturbation (ERSP) is the Time-Frequency analysis method of the ERP [34, 35]. ERSP analyzes the amplitude changes in the EEG signal's frequency spectrum as a function of time. The signal can be transformed in the Time-Frequency domain through several methods such as the STFT, Multi-Taper, Hilbert, and Wavelet Transforms.

Wavelets Analysis

The tracking of latency changes is an important aspect for ERP analysis, so the temporal information cannot be discarded. The Time-Frequency localization of wavelets — especially the Morlet wavelet — allows the transient and non-stationary EEG signals to be decomposed more effectively. For this reason the ERSP of an EEG signal was computed using the Continuous Wavelet Transform [2].

6.1.4 Single Trial Baseline Removal

A method for tracking the transient changes in the amplitudes and latencies of EEG signals is desired for neurosurgery and research. For example, the transient changes in the brain can identify the early stages of injury to the central nervous system when a patient is undergoing surgery. These methods require the background noise to be removed without the need of multiple trials.

The goal of most EEG analysis methods is to quantify and extract the response of a subject to a stimulus. To do this the response must be isolated from the brain's background neurological activity called the baseline. The baseline is the normal activity in the brain sustained throughout the experiment, and the signal is the change due to the stimulus.

The classical method of removing the baseline is by computing the Z-Score or standard score of the EEG signal [54, 55]. The standard score is defined as the number of standard deviations a given value is from the mean. The Z-Score is defined as

$$z_i = \frac{x_i - \mu}{\sigma}, \tag{6.2}$$

where μ is the mean and σ is the standard deviation of x. Traditionally the baseline is removed through a process called Multi-Trial Baseline Removal. This method computes the average Time-Frequency spectrograph for all of the trials, and then removes the baseline by calculating the Z-Score.

Multi-Trial Baseline Removal is defined as

$$ERSP_z(f,t) = \frac{ERP(f,t) - \mu_b(f)}{\sigma_b(f)}, \tag{6.3}$$

where: $ERP(f,t)$ is the average Event Related Potential response computed as a function of frequency and time for all trials, $\mu_b(f)$ and $\sigma_b(f)$ is the mean and variance of the spectral frequency f. The mean and variance are respectively computed as

$$\mu_b(f) = \frac{1}{mN} \sum_{k=1}^{N} \sum_{t=0}^{m-1} |F_k(f,t)|^2, \tag{6.4}$$

$$\sigma_b(f,k) = \sqrt{\frac{1}{mN-1} \sum_{t=0}^{m-1} \left(|F_k(f,t)|^2 - \mu_b'(f,t)\right)^2}. \tag{6.5}$$

Multi-Trial Baseline Removal computes the mean and variance over N trials, where k is the trial iterator and m is the number of samples before the stimulus. Like ERPs, Multi-Trial baseline removal requires many trials to isolate the stimulus response. As

a consequence, it has many of the same limitations of ERPs. For example, Multi-Trial baseline removal is known to be sensitive to noisy data trials [19]. As it was pointed out earlier, a method for removing the baseline trial by trial is required.

Single-Trial Baseline Removal removes the baseline from individual trials, and is performed by the following method

$$P(f,t) = \frac{|F_k(f,t)|^2 - \mu_b^{'}(f,k)}{\sigma_b^{'}(f,k)} \tag{6.6}$$

where: the power spectrum with the baseline removed is $P(f,t)$, and $F_k(f,t)$ is the spectral estimate as a function of frequency and time at trial k. The term $\mu_b^{'}$ is the mean of $F_k(f,t)$ given by

$$\mu_b^{'}(f,t) = \frac{1}{m}\sum_{t=0}^{m} |F_k(f,t)|^2 \tag{6.7}$$

The mean of the Single-Trial method is computed from the initial time of analysis until m, which is the number of samples before the stimulus. The term $\sigma_b^{'}(f,k)$ is the variance of $F_k(f,t)$ calculated as

$$\sigma_b^{'}(f,k) = \sqrt{\frac{1}{m-1}\sum_{t=0}^{m}\left(|F_k(f,t)|^2 - \mu_b^{'}(f,t)\right)^2} \tag{6.8}$$

Once the baseline is removed, the ERSP is computed by averaging over all of the trials given as

$$ERSP(f,t) = \frac{1}{N}\sum_{i=1}^{N} P_i(f,t). \tag{6.9}$$

Single-Trial Baseline Removal is used because it does not require multiple trials

to acquire the signal. It is more robust because it can handle noisy data, which is something that Multi-Trial Baseline Removal cannot.

6.1.5 ERSP Algorithm

The algorithm for computing the ERSP of the EEG signal with the single trial baseline removal is

1. For every trial:
   (a) Compute Algorithm 3.2
   (b) Remove the baseline from the CWT
   (c) Add the result to the ERSP
2. Divide the ERSP by the number of trials

6.1.6 Data Acquisition

In order to perform the algorithm, the signal must first be acquired from the experiments. EEG data is commonly stored in the BioSemi Data Format (BDF), or the European Data Format (EDF). In order to read these signals the `EDFLib` library was used. The computation of the ERSP requires the data from all of the stimuli to be known.

The data file is first opened using the EDFLib library. The data is then filtered for the stimuli represented as "triggers" located in the trigger channel. Once the location of the stimuli is known, the EEG information is copied into a contiguous memory block which contains all of the information from all of the triggers. The ERSP algorithm is computed on this data block.

6.1.7 Baseline Removal

The open source library GSL or GNU Scientific Library was used to remove the baseline from the signal [18]. The mean and variance were computed as outlined by equation 6.7 and 6.8 respectively. Where m is the pre-event count which is computed by multiplying the pre-event time t_p with the sampling frequency F_s

$$m = F_s t_p. \tag{6.10}$$

The baseline is then removed using the mean and variance equations outlined in equation 6.6. Finally, the result is divided over the number of trials to obtain the average.

6.1.8 Validation

A piecewise sinusoidal was analyzed to test the Single Trial Baseline Removal for the ERSP functions. The CWT of the signal

$$x(t) = \begin{cases} 2cos(\omega_0 t) & 1.5 \leq t \leq 2.0 \\ cos(\omega_0 t) & t < 1.5, t > 2.0 \end{cases} \tag{6.11}$$

was performed and the baseline of $cos(\omega_0 t)$ was removed at all times, while only the signal at 1.5 sec to 2.0 sec remained.

6.2 Mode Identification using Wavelet Transforms

Time-Frequency analysis was used to identify the modal parameters of an oscillating machine. This was accomplished by striking the machine and recording its vibrations

with a gyroscope. The modal parameters of the machine was analyzed by decomposing the impulse response using Time-Frequency analysis. This section will first review the Impulse Response, and Modal analysis. We then describe the method of analyzing a system's modal parameters by decomposing a signal's impulse response in the Time-Frequency domain.

6.2.1 Impulse Response

An impulse response is the output of a system when the input is an impulse [33]. An impulse is given as a Dirac delta function, defined by equation 2.2, in continuous time systems, and a Kronecker delta function, defined by equation 6.1, in discrete time systems [12]. The impulse functions contains every frequency, so it is a useful tool in signal processing for characterizing a system's behavior.

A system is mathematically modeled in controls systems engineering as linear functions. A function f is linear when it satisfies the homogeneity and additive properties respectively given as

$$f(au) = af(u), \tag{6.12}$$

$$f(u + v) = f(u) + f(v), \tag{6.13}$$

where a, u, and v are real numbers. A system is time invariant if the output of a system $g(t)$ is unchanged to the input $f(t)$ regardless of the delay of the input given as $f(t - \tau)$

$$g(t) = f(t) \Rightarrow g(t-\tau) = f(t-\tau).$$

A general linear time invariant (LTI) system $x(t)$ can be expressed as a function of impulses

$$x(t) = \int_{-\infty}^{+\infty} x(\tau)\delta(t-\tau)d\tau. \tag{6.14}$$

The unit impulse response of an LTI system $h(t)$ is the response to the unit impulse function $\delta(t)$

$$\delta(t) \rightarrow h(t). \tag{6.15}$$

Since the system is LTI, the following relationship holds

$$\delta(t-\tau) \rightarrow h(t-\tau).$$

Combining equations 6.14 and 6.15 and using the commutative property of the convolution, the output of a system from an impulse is

$$\begin{aligned} y(t) &= \int_{-\infty}^{+\infty} x(\tau)h(t-\tau)d\tau \\ &= \int_{-\infty}^{+\infty} x(t-\tau)h(\tau)d\tau \end{aligned}$$

This implies that the impulse response contains a complete input-output description of a system. When the impulse response is known, so is the output to any input [45].

6.2.2 Modal Analysis

The free response of a viscously dampened single degree of freedom system is

$$x(t) = Ae^{-\zeta\omega_n t}cos(\omega_d t + \theta) \tag{6.16}$$

where: ω_n is the undamped angular natural frequency, ζ is the dampening ratio, ω_d is the dampened natural frequency given as $\omega_d = \omega_n\sqrt{1-\zeta^2}$, and A is the magnitude. If it is assumed that the signal $x(t)$ is asymptotic — where the phase of the signal varies faster than the frequency — then the signal can be represented as

$$x(t) = A(t)cos(\delta(t)), \tag{6.17}$$

which is a function modulated in amplitude and frequency, where $A(t)$ and $\delta(t)$ are

$$A(t) = Ae^{-i\zeta\omega_n t} \tag{6.18}$$

$$\delta(t) = \omega_d t + \theta. \tag{6.19}$$

The Continuous Wavelet Transform can be used to identify the mechanical modal parameters of these types of systems [5, 32]. This can be used in mechanical engineering for identifying non trivial natural frequencies, dampening ratios, and settling times for complex systems. The Continuous Wavelet Transform of equation 6.17 is

$$\mathcal{W}_\phi(\tau, s) = \frac{\sqrt{s}}{2}A(\tau)\overline{\psi}(s\delta'(\tau))e^{i\delta(\tau)} \tag{6.20}$$

$$= \frac{\sqrt{s}}{2}Ae^{-\zeta\omega_n\tau}\overline{\psi}(s\omega_d)e^{i(\omega_d\tau+\theta)} \tag{6.21}$$

where: $\overline{\psi}$ is the conjugate complex of a wavelet class, and s is the scale of the wavelet. The absolute value of this CWT is at a maximum at a particular scale which is noted as s_0. The scale s_0 is correlated to the oscillating frequency of the system. This means that the CWT becomes

$$\mathcal{W}(\tau, s_0) = \frac{\sqrt{s_0}}{2} A e^{-\zeta\omega_n\tau}\overline{\psi}(s_0\omega_d)e^{i(\omega_d\tau+\theta)}. \tag{6.22}$$

This can then be simplified by taking the logarithm, the CWT then becomes

$$\ln|\mathcal{W}(\tau, s_0)| = -\zeta\omega_n\tau + \ln\left(\frac{\sqrt{s_0}}{2} A|\overline{\psi}(s_0\omega_d)|\right). \tag{6.23}$$

The linearity of the CWT means that an input signal with multiple modes such as

$$x(t) = \sum_{k=0}^{P-1} A_k(t) cos(\delta_k(t)) \tag{6.24}$$

would have a CWT of

$$\mathcal{W}_\phi(\tau, s) = \sum_{k=0}^{P-1} \frac{\sqrt{s_k}}{2} A_k(\tau)\overline{\psi}(s\delta_k'(\tau))e^{i\delta_k(\tau)}. \tag{6.25}$$

The dampening ratio ζ can be identified by the slope of the absolute logarithm of the CWT. The dampened frequency can be identified as $\omega_d = \omega_0/s_0$, and the settling time is identified using the algorithm outlined in section 6.2.3. The CWT of a system such as equation 6.24 is

$$\mathcal{W}_\phi(\tau, s_i) = \frac{\sqrt{s_i}}{2} A e^{-\zeta_i\omega_{n_i}\tau}\overline{\psi}(s_i\omega_{d_i})e^{i(\omega_{d_i}\tau+\theta_i)} \tag{6.26}$$

6.2.3 Mechanical Modal Analysis Algorithm

The algorithm to identify the settling time for the modal frequencies becomes

1. Compute Algorithm 3.2
2. Identify the local maximums for each frequency
3. Identify the peaks of the local maximums
4. For each peak
   (a) Determine the settling time

6.2.4 Identifying the Modal Parameters

To identify the modal signatures of a single degree of freedom system, an impulse is introduced, and its response is observed using the CWT. A mining machine was struck with a hammer, and its response was recorded from an accelerometer sampled at 500 Hz. To compute the modal parameters of the system, the wavelet transform is first computed (Figure 6.3).

Local maximums were identified at each frequency. A difference quotient was then computed on the local maximums to measure the rate of change. The peaks of the local maximums of the frequencies are when the difference quotient changes from a negative to a positive slope (Figure 6.4).

The settling times are then computed for the impulse response at the specific frequency. Settling time is defined as the time for the response to reach and stay within 2% of the final value [41]. Since an impact was introduced to a stationary system, the settling time is the time it takes for the system to return to its resting

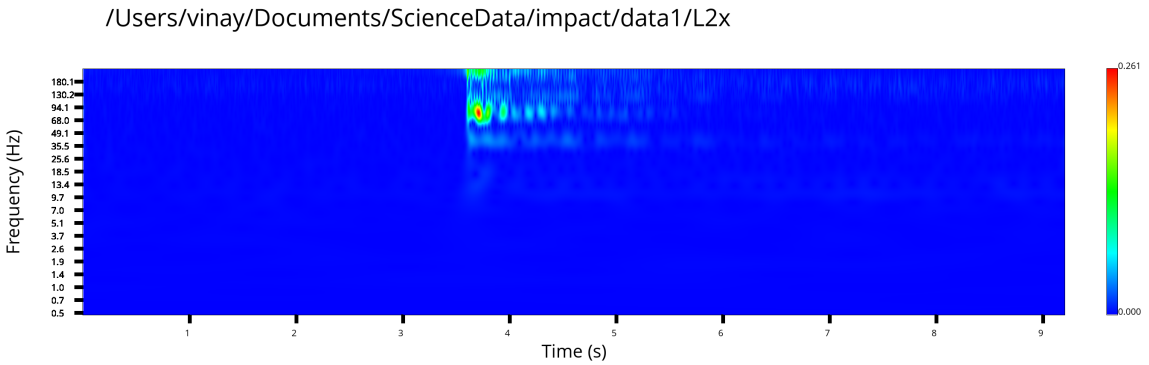


Figure 6.3: Vibrational response to an impact

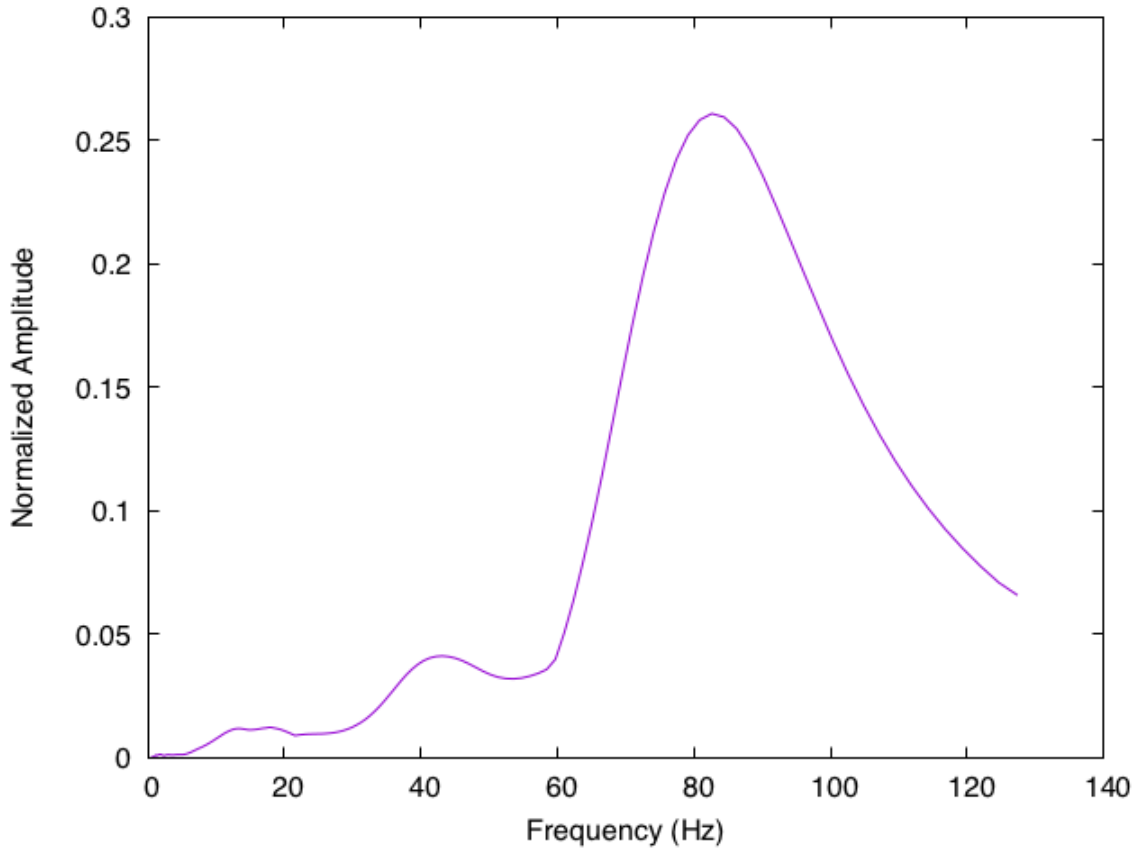


Figure 6.4: Local Maximums of the CWT at each frequency

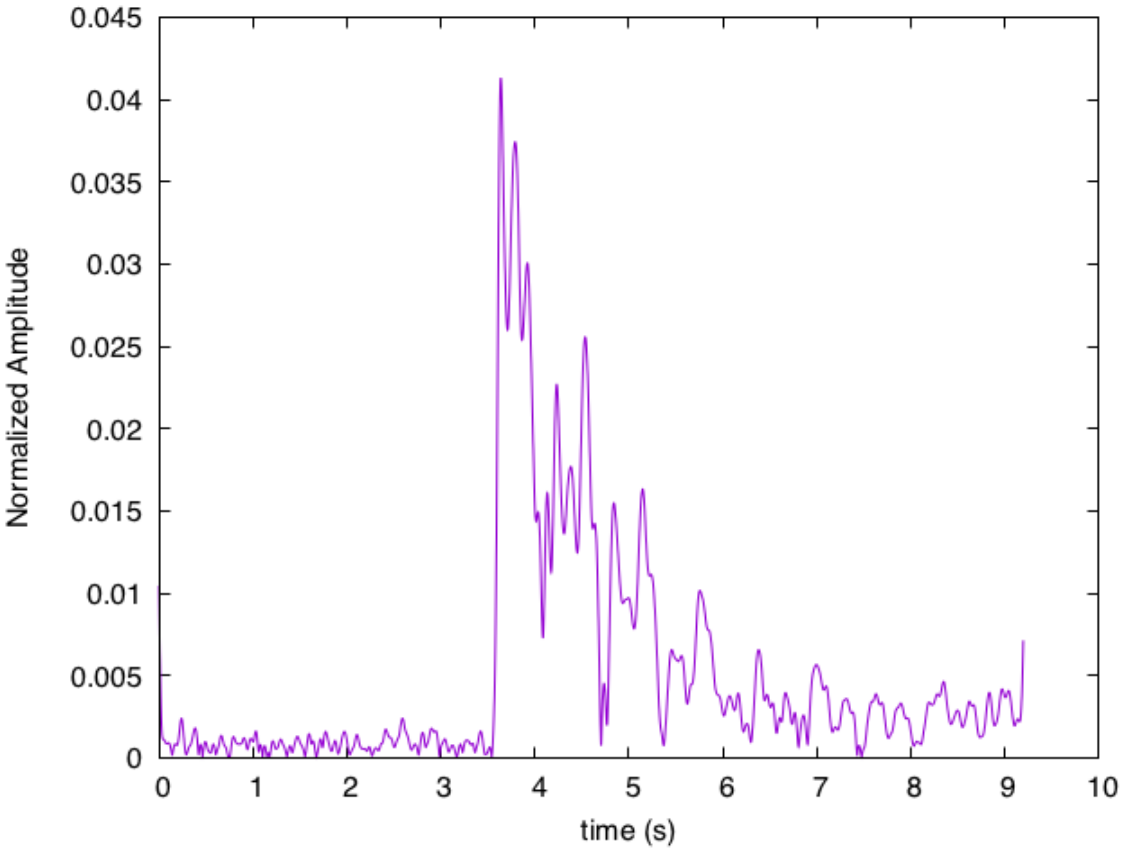


Figure 6.5: CWT of the impact at $f = 82.5Hz$

state. In this situation the settling time is the time it takes for the system to reach within 2% of the mean output before the impact (Figure 6.5). Table 6.2 shows the modal frequencies and settling times for the signal shown in Figure 6.3

Frequency (Hz)	Settling Time (s)
80.83 ± 3.50	$0.92 \pm 2.0 \times 10^{-3}$
42.20 ± 1.83	$1.06 \pm 2.0 \times 10^{-3}$
23.01 ± 1.00	$0.68 \pm 2.0 \times 10^{-3}$
12.82 ± 0.56	$0.50 \pm 2.0 \times 10^{-3}$

Table 6.2: Modal Parameters of an Oscillating Machine

Chapter 7

Concluding Remarks

The focus of this thesis was to study and implement methods of performing Time-Frequency analysis on a one-dimensional signal using the STFT and CWT. Through the process of developing the framework, many inconsistencies were found in the literature and nomenclature of the wavelet decomposition. As it was pointed out in section 3.2.1 the definitions of the wavelet functions, and the method of computing the wavelet transform were subject to change from article to article. As a consequence, the wavelet transform results differ depending on the implementation method and computational parameters used.

It would have been easier to utilize the pre-existing analysis functions found in MATLAB, however this would introduce several compromises. First, these functions are proprietary and require licensing, which limits its usability. Second, the MATLAB programs require more time to compute as it is a general solution, and not optimized for Time-Frequency analysis. Finally, the transform parameters, such as the wavelet's central frequency, are unknown. These parameters are vital for reducing error, and

improving speed and efficiency. While independently implementing the framework required some additional steps, it ensured that the correct Time-Frequency parameters were chosen and documented for the decomposition.

By implementing the framework in the C and C++ programming languages, we were able to utilize low level computational and memory manipulations to improve the speed while maintaining Time-Frequency accuracy. It also provides the user with various options such as speed versus detail, and `png` versus `gnuplot`. The STFT provides speed over detail, while the CWT provides detail over speed. The energy spectrum densities could be plotted as a `png` image, or on the interactive plotting library `gnuplot` for further manipulation.

Plotting the images were not trivial, as the Time-Frequency graphs are plotted linearly in the time domain, in a $\log_2$ scale in the frequency domain, and in a $\log_{10}$ scale in the normalized amplitude domain. Plotting the `png` image for example, required energy spectrums to be first converted into colours that represented low and high values, and then graphed pixel by pixel using the open source library `PNGwriter`. By doing this, we were able to generate a framework that is both functional, and free from proprietary licensing.

7.1 Future Work

The work on this project is by no means complete as there are additional tools that would help it improve its ability to decompose a signal in the Time-Frequency domain. For example, the framework can be expanded to compute different types of real and analytic wavelets rather than just the Morlet wavelet. One could also

increase the Time-Frequency options available by implementing additional decomposition methods such as the Winger-Ville distribution [59], as these methods have their own advantages.

In this framework, only the energy of the signal is analyzed. The framework could be expanded to analyze the phase information as well. Phase information can show the transient changes in the frequency tones. This could be advantageous when analyzing EEG signals to identify inter-trial coherence, which can show relations between different EEG trials.

There is also room for additional speedup and efficiency optimizations. While the framework is multi-threaded, the computation can be decreased significantly by using Graphics Processing Units (GPUs). Time-Frequency analysis has many "Single Instruction, Multiple Data" (SIMD) sections, which are more efficiently computed on GPUs.

Bibliography

[1] BP Abbott, Richard Abbott, TD Abbott, MR Abernathy, Fausto Acernese, Kendall Ackley, Carl Adams, Thomas Adams, Paolo Addesso, RX Adhikari, et al. Observation of gravitational waves from a binary black hole merger. *Physical review letters*, 116(6):061102, 2016.

[2] Paul S Addison. *The illustrated wavelet transform handbook: introductory theory and applications in science, engineering, medicine and finance*. CRC press, 2017.

[3] Uri M Ascher and Chen Greif. *A First Course on Numerical Methods*. SIAM, 2011.

[4] John Ashmead. Morlet wavelets in quantum mechanics. *arXiv preprint arXiv:1001.0250*, 2010.

[5] M Ashory, M Khatibi, M Jafari, and A Malekjafarian. Determination of mode shapes using wavelet transform of free vibration data. *Archive of Applied Mechanics*, 83(6), 2013.

[6] Salomon Bochner and Komaravolu Chandrasekharan. *Fourier Transforms.(AM-19)*, volume 19. Princeton University Press, 2016.

[7] CS Burrus, RA Gopinath, and H Guo. Generalizations of the basic multiresolution wavelet system. *Introduction to Wavelets and Wavelet Transforms: A Primer. Prentice-Hall, Inc*, pages 98–145, 1997.

[8] Thomas Christensen. *The Cambridge history of Western music theory*. Cambridge University Press, 2006.

[9] Charles K Chui. *An introduction to wavelets*. Elsevier, 2016.

[10] Leon Cohen. *Time-frequency analysis*, volume 778. Prentice Hall PTR Englewood Cliffs, NJ:, 1995.

[11] James W Cooley and John W Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[12] Paul Adrien Maurice Dirac. *The principles of quantum mechanics*. 27. Oxford university press, 1981.

[13] Robert Edelberg. Biopotentials from the skin surface: The hydration effect. *Annals of the New York Academy of Sciences*, 148(1):252–262, 1968.

[14] Marie Farge. Wavelet transforms and their applications to turbulence. *Annual review of fluid mechanics*, 24(1):395–458, 1992.

[15] Qiang Feng and Bing-Zhao Li. Convolution theorem for fractional cosine-sine transform and its application. *Mathematical Methods in the Applied Sciences*, 40(10):3651–3665, 2017. mma.4251.

[16] Matteo Frigo and Steven G Johnson. Fftw: An adaptive software architecture for

the fft. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 3, pages 1381–1384. IEEE, 1998.

[17] Dennis Gabor. Theory of communication. part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, 93(26):429–441, 1946.

[18] Brian Gough. *GNU scientific library reference manual*. Network Theory Ltd., 2009.

[19] Romain Grandchamp and Arnaud Delorme. Single-trial normalization for event-related spectral decomposition reduces sensitivity to noisy trials. *Frontiers in psychology*, 2:236, 2011.

[20] Alexander Grossmann and Jean Morlet. Decomposition of hardy functions into square integrable wavelets of constant shape. *SIAM journal on mathematical analysis*, 15(4):723–736, 1984.

[21] Michael T.. Heath. *Scientific computing: An introductory survey*. McGraw-Hill, 1997.

[22] Hwei P. Hsu. *Schaum's Outline of Signals and Systems, Third Edition*. McGraw-Hill Education: New York, Chicago, San Francisco, Athens, London, Madrid, Mexico City, Milan, New Delhi, Singapore, Sydney, Toronto, 2014.

[23] Barbara Burke Hubbard. *The World According to Wavelets The Story of a Mathematical Technique in the Making*. Universities Press, 2005.

[24] Anders Isaksson, Arne Wennberg, and Lars H Zetterberg. Computer analysis

of eeg signals with parametric models. *Proceedings of the IEEE*, 69(4):451–461, 1981.

[25] Philipp K Janert. Gnuplot in action. *Understanding data with Graphs. 2nd ed. Greenwich: Manning Publications*, 2009.

[26] Maarten Jansen. *Noise reduction by wavelet thresholding*, volume 161. Springer Science & Business Media, 2012.

[27] Philip Kiameh. *Electrical Equipment Handbook: Troubleshooting and Maintenance: Troubleshooting and Maintenance*. McGraw Hill Professional, 2003.

[28] L.G. Kiloh, A.J. McComas, J.W. Osselton, and A.R.M. Upton. *Clinical Electroencephalography*. Butterworths & Co Ltd., fourth edition, 1981.

[29] Richard Kronland-Martinet, Jean Morlet, and Alexander Grossmann. Analysis of sound patterns through wavelet transforms. *International Journal of Pattern Recognition and Artificial Intelligence*, 1(02):273–302, 1987.

[30] Myer Kutz. *Standard handbook of biomedical engineering and design*. McGraw-Hill, 2003.

[31] Myer Kutz. *Biomedical Engineering and Design Handbook, Volume 2: Volume 2: Biomedical Engineering Applications*, volume 2. McGraw Hill Professional, 2009.

[32] Joseph Lardies and Stephane Gouttebroze. Identification of modal parameters using the wavelet transform. *International Journal of Mechanical Sciences*, 44(11):2263–2283, 2002.

[33] Edward A Lee. *Structure and interpretation of signals and systems*. Lee & Seshia, 2011.

[34] Scott Makeig. Auditory event-related dynamics of the eeg spectrum and effects of exposure to tones. *Electroencephalography and clinical neurophysiology*, 86(4):283–293, 1993.

[35] Scott Makeig, Stefan Debener, Julie Onton, and Arnaud Delorme. Mining event-related brain dynamics. *Trends in cognitive sciences*, 8(5):204–210, 2004.

[36] Stéphane Mallat. *A wavelet tour of signal processing*. Academic press, 1999.

[37] Stephane Mallat and Sifen Zhong. Characterization of signals from multiscale edges. *IEEE Transactions on pattern analysis and machine intelligence*, 14(7):710–732, 1992.

[38] Steven D Meyers, Brian G Kelly, and James J O'Brien. An introduction to wavelet analysis in oceanography and meteorology: With application to the dispersion of yanai waves. *Monthly weather review*, 121(10):2858–2866, 1993.

[39] Saraju P Mohanty. *Nanoelectronic mixed-signal system design*. McGraw-Hill Education, 2015.

[40] Douglas Myers. File:eeg cap.jpg — wikipedia, the free encyclopedia, 2010. [Online; accessed 24-September-2017].

[41] Norman S Nise. *CONTROL SYSTEMS ENGINEERING, (With CD)*. John Wiley & Sons, 2007.

[42] Antonia Papandreou-Suppappola. *Applications in time-frequency signal processing*, volume 10. CRC press, 2002.

[43] Sybil P Parker. *McGraw-Hill dictionary of scientific and technical terms*. McGraw-Hill, 1984.

[44] Arno A Penzias and Robert Woodrow Wilson. A measurement of excess antenna temperature at 4080 mc/s. *The Astrophysical Journal*, 142:419–421, 1965.

[45] Charles L Phillips, John M Parr, and Eve A Riskin. *Signals, systems, and transforms*. Prentice Hall, 1995.

[46] Phillip James Plauger. *The standard C library*. Prentice Hall PTR, 1991.

[47] Howard Percy Robertson. The uncertainty principle. *Physical Review*, 34(1):163, 1929.

[48] Ulrich L. Rohde, Jerry C. Whitaker, and Hans Zahnd. *Communications receivers: principles and design*. McGraw-Hill Education, 2017.

[49] Steven W Smith et al. *The scientist and engineer's guide to digital signal processing*. California Technical Pub. San Diego, 1997.

[50] Juliu O Smith III. *Spectral audio signal processing*. W3K publishing, 2011.

[51] Leif Sörnmo and Pablo Laguna. *Bioelectrical signal processing in cardiac and neurological applications*, volume 8. Academic Press, 2005.

[52] Hans-Georg Stark. *Wavelets and signal processing: an application-based introduction*. Springer Science & Business Media, 2005.

[53] C. Sujatha. *Vibration and Acouistic Measurement and Signal Analysis*. McGraw-Hill Professional, 2010.

[54] Catherine Tallon-Baudry, Olivier Bertrand, Claude Delpuech, and Jacques Pernier. Stimulus specificity of phase-locked and non-phase-locked 40 hz visual responses in human. *Journal of Neuroscience*, 16(13):4240–4249, 1996.

[55] Catherine Tallon-Baudry, Andreas Kreiter, and Olivier Bertrand. Sustained and transient oscillatory responses in the gamma and beta bands in a visual short-term memory task in humans. *Visual neuroscience*, 16(3):449–459, 1999.

[56] John Teslade. File:stft - windows.png — wikipedia, the free encyclopedia, 2005. [Online; accessed 24-September-2017].

[57] Christopher Torrence and Gilbert P Compo. A practical guide to wavelet analysis. *Bulletin of the American Meteorological society*, 79(1):61–78, 1998.

[58] Michael Unser, Akram Aldroubi, and Murray Eden. On the asymptotic convergence of b-spline wavelets to gabor functions. *IEEE transactions on information theory*, 38(2):864–872, 1992.

[59] Eugene Wigner. On the quantum correction for thermodynamic equilibrium. *Physical review*, 40(5):749, 1932.

[60] Thoru Yamada and Elizabeth Meng. *Practical guide for clinical neurophysiologic testing: EEG*. Lippincott Williams & Wilkins, 2012.