3 Time-Frequency Signal Analysis

Time-dependent signals are functions with time as their independent variable. Time-dependent signals could be temperature logs, stock-index profiles, electrocardiogram signals, vibration signals, and speech signals. To understand the physics and the underlying behavior and patterns of these signals, implementing signal analysis tools is needed. The Fourier transform is one of the most important and foundational methods for the analysis of time signals. However, it was realized very early on that Fourier transform-based methods had severe limitations. Specifically, when transforming a given time signal, it is clear that all the frequency content of the signal can be captured with the transform, but the transform fails to capture the moment in time when various frequencies were exhibited. Fig(25) shows a prototypical signal $S(t)$ that is comprised of various frequency components exhibited at different times. For instance, at the beginning of the signal, there are high-frequency components whereas in the middle, there are relatively low-frequency oscillations. The Fourier transform of the signal contains all this information, but there is no indication of *when* the high or low frequencies occur in time [5]. This section overcomes this limitation by going through several Fourier-modification methods that allow for the simultaneous extraction of time and frequency information.

3.1 Fourier Series & Fourier Transform

A central concern of mathematical physics and engineering mathematics involves the transformation of equations into a coordinate system where expressions simplify, decouple, and are amendable to computation and analysis [4]. Perhaps the most foundational and ubiquitous coordinate transformation was introduced by J. B. Joseph Fourier in the early 1800s to investigate the theory of heat [15]. Fourier introduced the concept that sine and cosine functions of increasing frequency provide an orthogonal basis for the space of solution functions. Before going through the limitations of this concept, it would be helpful to cover the basics of Fourier series and Fourier transforms so we can build upon this knowledge for later modifications.

3.1.1 Fourier Series

A fundamental result in Fourier analysis is that if $f(x)$ is a periodic and piecewise smooth function, then it can be written in terms of a Fourier series, which is an infinite sum of cosines and sines of increasing frequencies. In particular, if $f(x)$ is 2π-periodic, it can be written as:

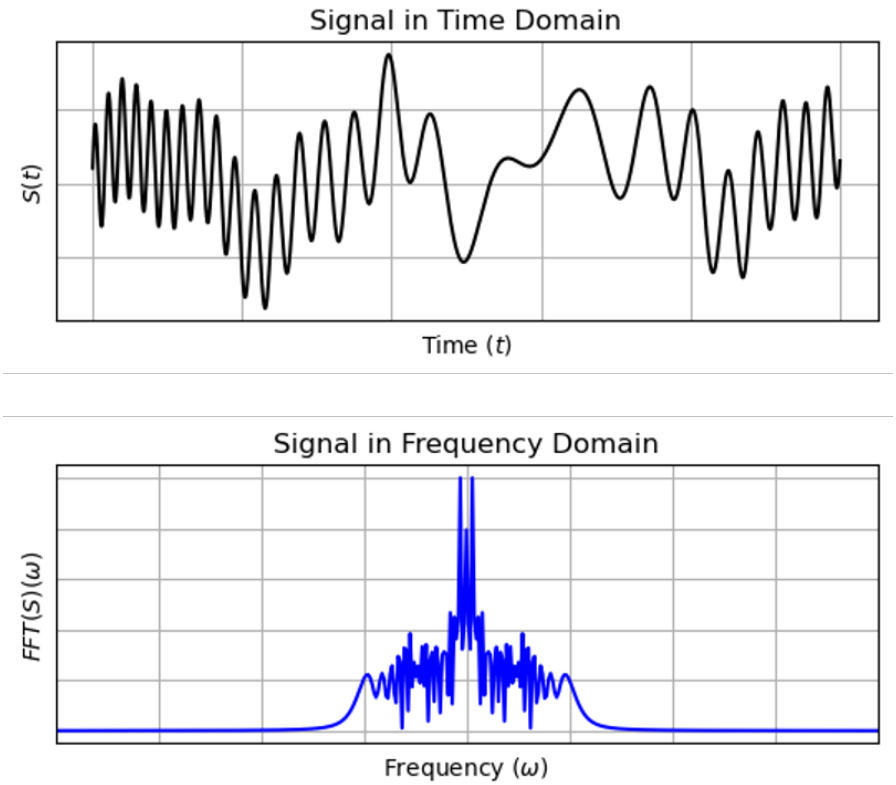

Figure 25: Signal $S(t)$ and its normalised Fourier Transform.

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left[a_n \cos(nx) + b_n \sin(nx) \right] \tag{53}$$

The coefficients a_0, a_n, and b_n are given by:

$$a_0 = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \mathrm{d}x \tag{54a}$$

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(nx) \mathrm{d}x \tag{54b}$$

$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin(nx) \mathrm{d}x \tag{54c}$$

As shown in Fig(26), the signal $f(x)$ can be broken down to its basic frequency components of sines and cosines.
The main formula expressed in Eq.(26) can be further simplified when dealing with odd or even functions. As a reminder, a function $f(x)$ is said to be even if

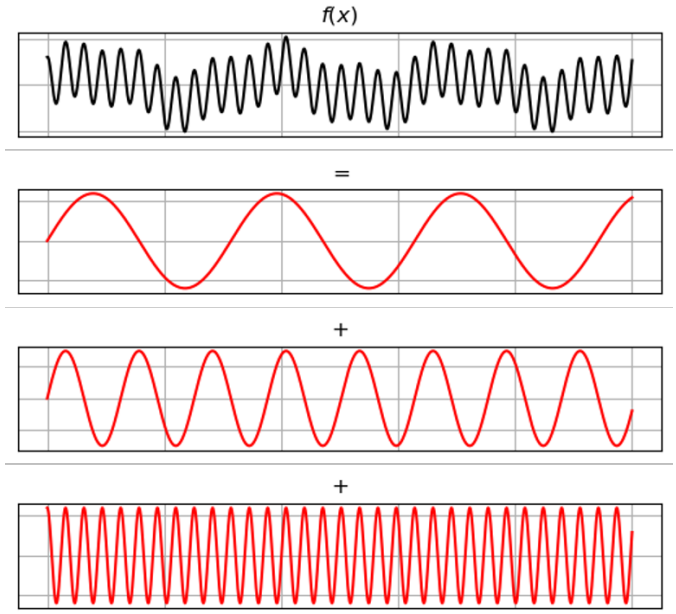

Figure 26: The function $f(x)$ is broken into its basic frequency components of sines and cosines.

$f(-x) = f(x)$, , for all real numbers x, and its integral over the whole domain gives:

$$\int_{-\pi}^{\pi} f(x)\mathrm{d}x = 2\int_{0}^{\infty} f(x)\mathrm{d}x$$

For even functions, the following can be applied:

- $f(x)\cos(nx)$ is even and hence $a_n = \dfrac{2}{\pi}\displaystyle\int_0^{\pi} f(x)\cos(nx)\mathrm{d}x$,
- $f(x)\sin(nx)$ is even and hence $b_n = 0$

On the other hand, the function $f(x)$ is said to be odd if $f(-x) = -f(x)$, for all real numbers x, and its integral over the whole domain gives:

$$\int_{-\pi}^{\pi} f(x)\mathrm{d}x = 0$$

For odd functions, the following can be applied:

- $f(x)\cos(nx)$ is odd and hence $a_n = 0$,

- $f(x)\sin(nx)$ is even and hence $b_n = \frac{2}{\pi}\int_0^{\pi} f(x)\sin(nx)\mathrm{d}x$

Graphically, as shown in Fig(27), even functions have symmetry about the y-axis, whereas odd functions have symmetry around the origin.

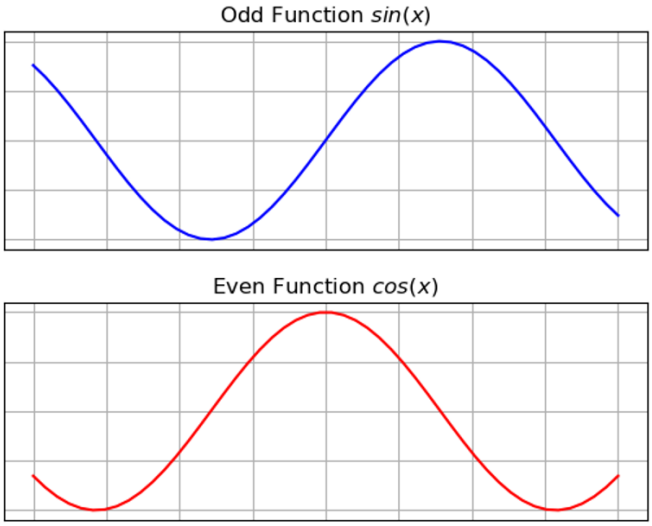

Figure 27: Odd and Even Functions.

To find a Fourier series, it is sufficient to calculate the integrals that give the coefficients a_0, a_n, and b_n and plug them into the big series formula.
It is helpful (especially for the next sections) to rewrite the trigonometric Fourier series representation in Eq.(53)as a complex exponential Fourier series. To do this, Euler's formulae are used:

$$\cos(nx) = \frac{e^{inx} + e^{-inx}}{2} \qquad \sin(nx) = \frac{e^{inx} + e^{-inc}}{2i}$$

Then, we can write the following:

$$
\begin{aligned}
f(x) &= \frac{a_0}{2} + \sum_{n=1}^{\infty} \left[a_n \cos(nx) + b_n \sin(nx) \right] \\
&= \frac{a_0}{2} + \sum_{n=1}^{\infty} \left[a_n \left(\frac{e^{inx} + e^{-inx}}{2} \right) + b_n \left(\frac{e^{inx} - e^{-inx}}{2i} \right) \right] \\
&= \frac{a_0}{2} + \sum_{n=1}^{\infty} \left[\frac{a_n e^{inx} + a_n e^{-inx}}{2} + \frac{b_n e^{inx} - b_n e^{-inx}}{2i} \right] \\
&= \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(\frac{a_n - ib_n}{2} \right) e^{inx} + \sum_{n=1}^{\infty} \left(\frac{a_n + ib_n}{2} \right) e^{-inx}
\end{aligned}
$$

The coefficients of the complex exponentials can be rewritten by defining $c_n^* = \frac{a_n - ib_n}{2}$ and $c_n = \frac{a_n + ib_n}{2}$ where $n = 1, 2, \cdots$, and the constant as $c_0 = \frac{a_0}{2}$. This gives:

$$
f(x) = c_0 + \sum_{n=1}^{\infty} c_n^* e^{inx} + \sum_{n=1}^{\infty} c_n e^{-inx}
$$

To simplify this sum, it would be practical to make the exponentials in both sums the same. This way, we can add all three terms under only one sum. This can be done by re-indexing the first sum by replacing $n = -n$. This necessarily changes the limits of the sum to be:

$$
f(x) = c_0 + \sum_{n=-1}^{-\infty} c_{-n}^* e^{-inx} + \sum_{n=1}^{\infty} c_n e^{-inx}
$$

If we define $c_{-n}^* = c_n$ where $n = -1, -2, \cdots$, then we can now combine all of the terms into a simple sum:

$$
f(x) = \sum_{n=-\infty}^{\infty} c_n e^{-inx} \tag{55}
$$

where the coefficient can be computed as:

$$
c_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x) e^{inx} \mathrm{d}x
$$

3.1.2 Fourier Transform (FT)

The Fourier series is used to represent a periodic function, $x \in [-\pi, \pi]$, by a discrete sum of complex exponentials. In contrast, the Fourier transform is used

to represent **a general, non-periodic function** by a continuous superposition or integral of complex exponentials. The Fourier transform can also be viewed as the limit of the Fourier series of a function with the period approaching infinity, so the **limits of integration change from one period to** $x \in [-\infty, \infty]$. The Fourier transform and its inverse are given by:

$$F(k) = \int_{-\infty}^{\infty} e^{-ikx} f(x) \mathrm{d}x \tag{56a}$$

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{ikx} F(k) \mathrm{d}k \tag{56b}$$

where $f(x)$ is the function and $F(k)$ is its Fourier transform with k being the frequency variable. These integrals are known as the Fourier transform pair.

3.1.3 Fourier Transform and Derivatives

The critical property in the usage of the Fourier transform concerns derivatives relations. To see how these properties are generated, we begin by considering the Fourier transform of $f'(x)$. We denote the Fourier transform of $f(x)$ as $\widehat{f(x)}$. Thus we find [5]:

$$\widehat{f'(x)} = \int_{-\infty}^{\infty} e^{-ikx} f'(x) \mathrm{d}x$$

Let's solve this integral by parts where $\mathrm{d}v = f'(x) \implies v = f(x)$ and $u = e^{-ikx} \implies du = -ike^{-ikx}\mathrm{d}x$. Therefore, the integral can be written as:

$$\begin{aligned}
\widehat{f'(x)} &= \left[e^{-ikx} f(x) \Big|_{-\infty}^{+\infty} - \int_{-\infty}^{+\infty} f(x)(-ik)e^{-ikx} \mathrm{d}x \right] \\
&= \left[e^{-ikx} f(x) \Big|_{-\infty}^{+\infty} + (ik) \int_{-\infty}^{+\infty} f(x) e^{-ikx} \mathrm{d}x \right]
\end{aligned}$$

If $f(x)$ is periodic, then the term $e^{-ikx} f(x) \Big|_{-\infty}^{+\infty} = 0$, thus we get:

$$\widehat{f'(x)} = \left[(ik) \int_{-\infty}^{+\infty} f(x) e^{-ikx} \mathrm{d}x \right] \tag{57}$$

$$= (ik)\widehat{f(x)} \tag{58}$$

It is easy to generalize this argument to an arbitrary number of derivatives. The final result is the following relation between the Fourier transform of the derivative and the Fourier transform itself:

$$\widehat{f^{(n)}} = (ik)^n \widehat{f} \tag{59}$$

This property is an essential property when it comes to solving differential equations and what makes the Fourier transform so useful and practical.

3.1.4 Discrete Fourier Transform (DFT)

Until now, we have considered the Fourier series and Fourier transform for continuous functions $f(x)$. However, when working with real data, we don't have an analytic function; we have data measurements from experiments or simulations, i.e., discrete points. Therefore, it is necessary to approximate the Fourier transform on a discrete vector of data. The resulting truncated form is known as the discrete Fourier transform (DFT), which is a mathematical transformation that can be written in terms of a big matrix multiplication. Although we will always use the fast Fourier transform (FFT) for computations, it is illustrative to begin with the simplest formulation of DFT. To illustrate the idea of DFT, suppose that data measurements $f(x_0), f(x_1), \cdots, f(x_{n-1})$ are captured at certain discrete points $x_0, x_1, \cdots, x_{n-1}$ with regular spacing Δx. Hence, we have a vector of data $\mathbf{f} = [f_0, f_1, \cdots, f_{n-1}]^T$, where $f_n = f(x_n)$, which represents the captured measurements, as shown in Fig(28). To Fourier analyze this vector, we need to compute its Fourier transform components, i.e., $\hat{\mathbf{f}} = [\hat{f}_0, \hat{f}_1, \cdots, \hat{f}_{n-1}]$.

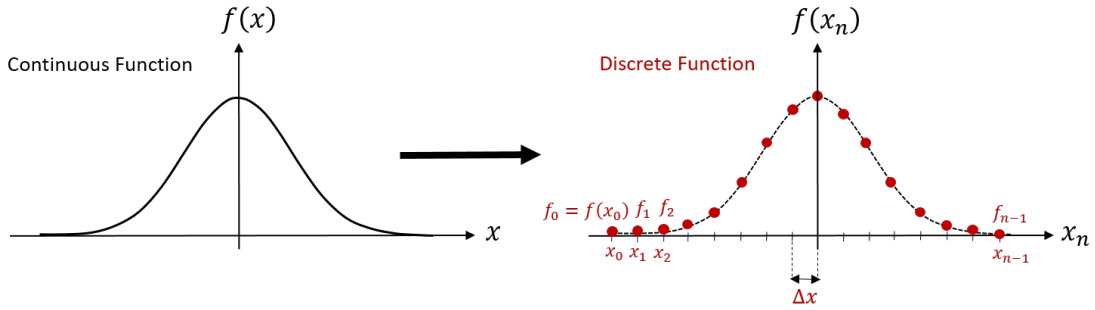

Figure 28: Discrete Data Sampled for the Discrete Fourier Transform.

The discrete Fourier transform that can be used to analyze such a discrete data vector is given by:

$$\hat{f}_k = \sum_{j=0}^{n-1} f_j e^{-\frac{i2\pi}{n}jk} \tag{60a}$$

$$f_j = \frac{1}{n}\sum_{k=0}^{n-1} \hat{f}_k e^{\frac{i2\pi}{n}jk} \tag{60b}$$

This is known as the discrete Fourier transform pair. Thus, the DFT is a linear operator (i.e., a matrix) that maps the data points in $\mathbf{f}$ to the frequency domain $\hat{\mathbf{f}}$:

$$\{f_0, f_1, \cdots, f_{n-1}\} \qquad \underrightarrow{\text{DFT}} \qquad \left\{\hat{f}_0, \hat{f}_1, \cdots, \hat{f_{n-1}}\right\}$$

For a given number of points n, the DFT represents the data using sine and cosine functions with integer multiples of a fundamental frequency $\omega_n = e^{-\frac{2\pi i}{n}}$. The DFT may be, hence, computed by the matrix multiplication:

$$\begin{bmatrix} \hat{f}_0 \\ \hat{f}_1 \\ \hat{f}_2 \\ \vdots \\ \hat{f_{n-1}} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_n & \omega_n^2 & \cdots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \cdots & \omega_n^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \cdots & \omega_n^{(n-1)^2} \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{n-1} \end{bmatrix} \tag{61}$$

The matrix is called the DFT matrix, and it is a complex matrix that computes the frequency components of discrete measurements of data points. Here, the left-hand side vector $\hat{\mathbf{f}}$ contains all the frequency components that are in the measurements vector $\mathbf{f}$, for example, $\hat{f}_1$ tells us how much of the first low frequency is in the data, $\hat{f}_2$ tells us how much of the next higher frequency is in the data, etc. Fig(29) shows the real part of the DFT matrix for $n = 256$. It can be seen from this image that there is a hierarchical and highly symmetric multiscale structure of this matrix. Each row and column is a cosine function with increasing frequency.
The DFT is tremendously useful for numerical approximation and computation, but it does not scale well to very large $n \gg 1$, as the simple formulation involves multiplication by a dense $n \times n$ matrix, requiring $\mathcal{O}(n^2)$ operations.

3.1.5 Fast Fourier Transform (FFT)

In the mid-1960s, Cooley and Tukey developed what is now commonly known as the fast Fourier transform, the FFT algorithm [16]. The FFT is a clever way to compute the Discrete Fourier Transform (DFT) much faster by reusing computations instead of repeating them. The key idea is that instead of directly computing

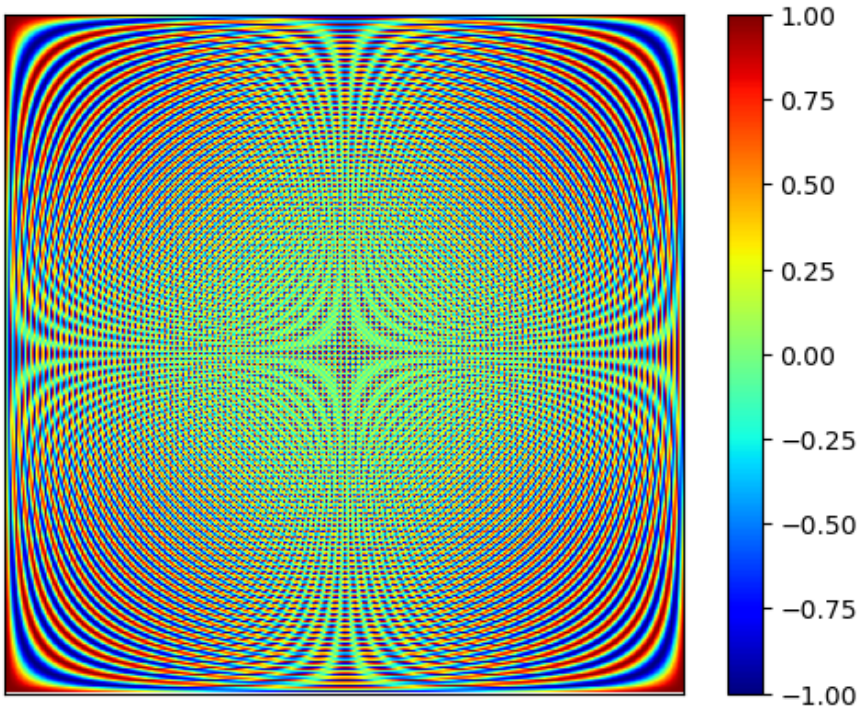

Figure 29: Real part of DFT matrix for $n = 256$.

all frequency components from all data points at once, the FFT breaks the problem into smaller pieces using the so-called: butterfly structure. The algorithm was named one of the top ten algorithms of the twentieth century for one reason: **the operation count for solving a system dropped to** $\mathcal{O}(n\log(n))$. Simply put, a 1D FFT takes a problem of size n, cuts it into two problems of size $n/2$, cuts the two into four problems of size $n/4$, cuts the four into eight problems of size $n/8$, all the way down to n problems of size 1, which is very easy to solve. Going this far takes n operations to compute. The recursion depth to go back to the original problem requires $\log(n)$ operations. Therefore, the total complexity becomes $\mathcal{O}(n\log(n))$. As mentioned earlier, multiplying by the DFT matrix involves $\mathcal{O}(n^2)$ operations. The fast Fourier transform, as it scales to $\mathcal{O}(n\log(n))$, enables a tremendous range of applications, including audio and image compression, streaming video, satellite communications, and the cellular network, to name only a few of the myriad applications [4]. For example, audio is generally sampled at 44.1kHz, or $44,100$ samples per second. For 10 seconds of audio, the vector $\mathbf{f}$ will have dimension $n = 4.41 \times 10^5$. Computing the DFT using matrix multiplication involves approximately 2×10^{11}, or 200 billion, multiplications. In contrast, the FFT requires approximately 6×10^6, which amounts to a speed-up factor of

over $30,000$. **Very important to note here that the FFT's basic idea is that the DFT may be implemented much more efficiently if the number of data points n is a power of 2, i.e., 2^n.**
To gain familiarity with how to use and interpret the FFT, we will begin with a simple example that uses the FFT to denoise a signal. Consider a time-dependent function $f(t)$ given by:

$$f(t) = 2\cos(2\pi f_1 t) + \sin(2\pi f_2 t)$$

with frequencies $f_1 = 50$ and $f_2 = 120$. We then add a large amount of Gaussian white noise to this signal, as shown in the top panel of Fig(30). It is possible to compute the fast Fourier transform of this noisy signal using the `fft` command (it is the same command in Python and in Matlab). The power spectral density (PSD) is the normalized squared magnitude of $\hat{\mathbf{f}}$, and indicates how much power the signal contains in each frequency. The middle panel in Fig(30) shows clearly that the noisy signal contains two large peaks at 50Hz and 120Hz. It is possible to zero out components with power below a threshold to remove noise from the signal. After inverse transforming the filtered signal, we find the clean and filtered time-series match quite well, in the bottom panel of Fig(30).

3.2 Windowed Fourier Transform

Although the Fourier transform provides information about the frequency content of a given signal, it does not give any information about *when* those frequencies occur. For a signal with non-stationary frequency content, such as a musical composition, it is important to characterise the frequency content and its evolution in time simultaneously.

The limitations of the direct application of the Fourier transform, and its inability to localize a signal in both the time and frequency domains, were realized very early in the development of radar and sonar detection. The Hungarian physicist, mathematician, and electrical engineer G*á*bor D*é*nes (Nobel Prize for Physics in 1971 for the discovery of holography in 1947) was the first to propose a formal method for localizing both time and frequency. The G*á*bor transform, also known as the short-time Fourier transform (STFT), involves a simple modification of the Fourier transform kernel, which is given by:

$$g_{t,\omega}(\tau) = e^{i\omega\tau} g(t-\tau)$$

where the new term to the Fourier kernel $g(t-\tau)$ was introduced to localize both time and frequency. The G*á*bor transform is defined as the following:

$$\mathcal{G}[f](t,\omega) = \hat{f}_g(t,\omega) = \int_{-\infty}^{\infty} f(\tau)\bar{g}(t-\tau)e^{-i\omega\tau}\mathrm{d}\tau = \langle f(\tau), \bar{g}(\tau)_{t,\omega}\rangle \qquad (62)$$

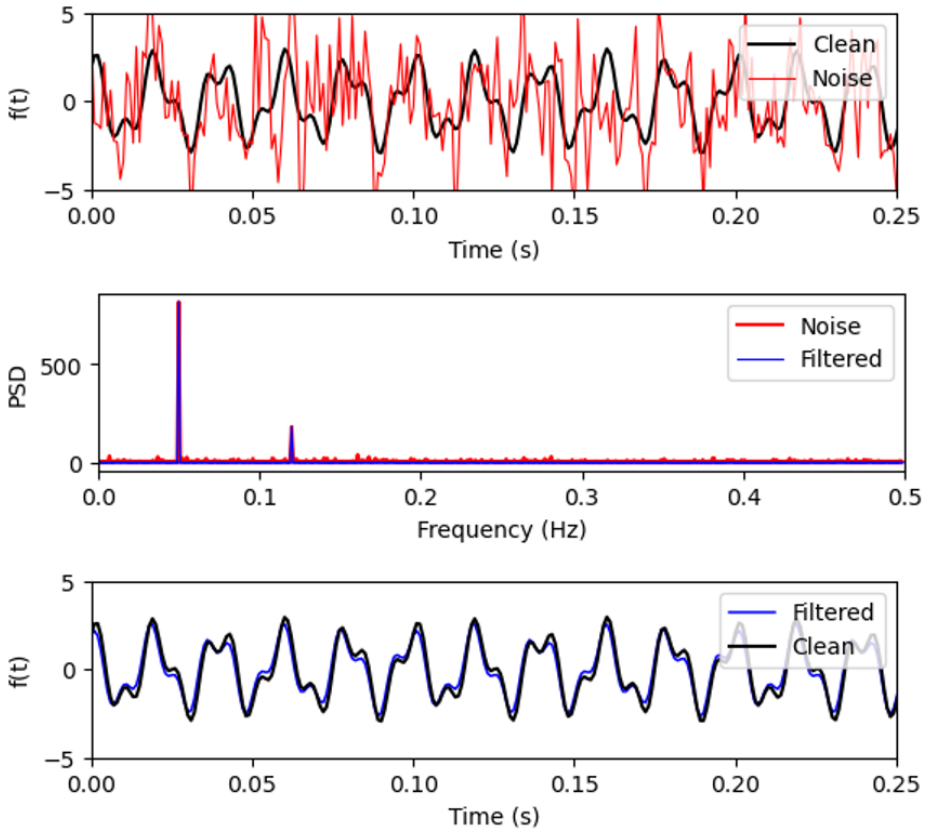

Figure 30: Denoising with FFT. (top) Noise is added to a simple signal given by a sum of two waves. (middle) In the Fourier domain, dominant peaks may be selected, and the noise is filtered. (bottom) The de-noised signal is obtained by inverse Fourier transforming the two dominant peaks.

where the bar denotes the complex conjugate of the function. The Eq.(31) shows that the Gábor transform is a function of two variables; time t and frequency ω. Thus, the function $g_{t,\omega}(\tau)$ acts as a time filter for localizing the signal over a specific time window. The integration over the parameter τ slides the time-filtering window down the entire signal to pick out the frequency information at each instant of time. The function $g(t)$ is the kernel, and is often chosen to be a Gaussian:

$$g(t) = e^{-(t-\tau)^2/a^2}$$

Here the parameter a determines the spread (or the width) of the short-time window for the Fourier transform, and τ determines the center of the moving window. For small values of a, the resolution in time is high while the resolution

in frequency is low. Conversely, for large values of a, the frequency resolution improves while the time resolution decreases. This is illustrated in Fig(31) as it shows in the top panel the time-varying signal $f(t)$, in black, and the moving Gábor Gaussian window, in red. The middle panel shows the convolution between the signal and the Gábor function. The Fourier transform of the resulting signal is shown in the bottom panel.

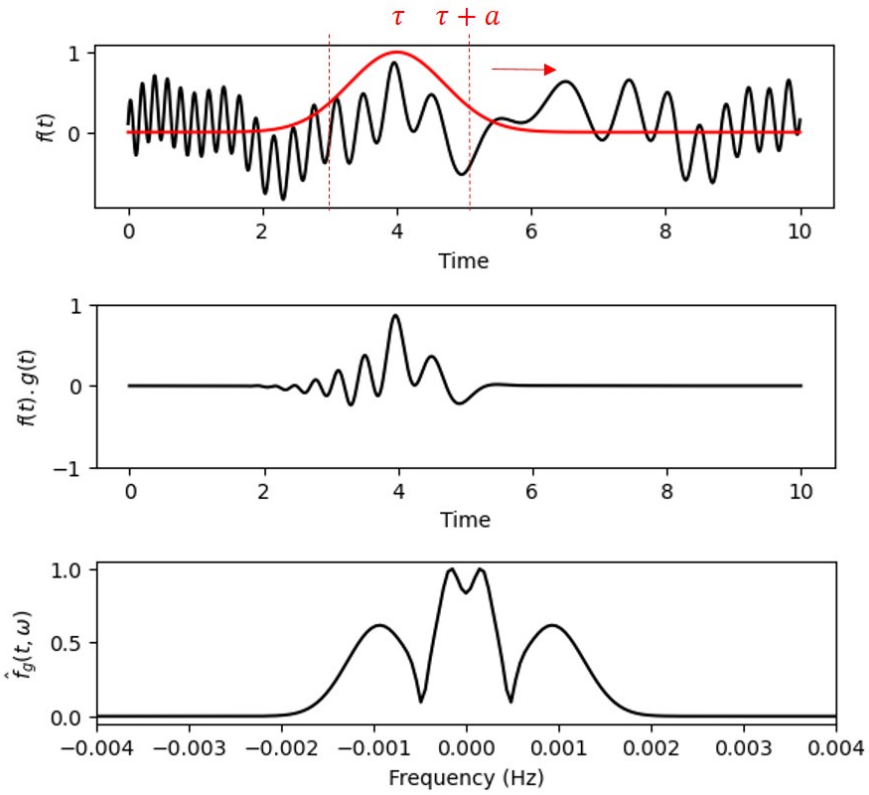

Figure 31: Time signal $f(t)$ and the Gábor time filter $g(t) = \exp\left(-(t-\tau)^2/a^2\right)$ where $\tau = 4$ and $a = 1$ are shown in the top panel. The product $f(t) * g(t)$ is shown in the middle panel and its Fourier transform $\hat{f}_g(t, \omega)$ in the bottom panel.

3.3 Spectrogram

A spectrogram is a visual representation of the spectrum of frequencies of a signal and their amplitudes as they vary with time, all on one graph. Fig(32) shows the spectrogram, top panel, generated from the time-varying signal $f(t)$, bottom panel. Spectrograms are two-dimensional graphs, with a third dimension represented by colors. Time runs from left (oldest) to right (youngest) along the horizontal axis.

The vertical axis represents frequency, with the lowest frequencies at the bottom and the highest at the top. The amplitude (or energy or "loudness") of a particular frequency at a specific time is represented by the third dimension, color, with dark reds corresponding to low amplitudes and brighter colors up through blue corresponding to progressively stronger (or louder) amplitudes. The Spectrogram matrix, where its rows correspond to the frequencies captured at a specific time and its columns correspond to the time length of the signal, is computed by taking the absolute value of the Gábor transform. Spectrograms are most useful when you want to see how the frequency content of a signal changes over time — especially for signals that are non-stationary (their frequency content is not constant). They are commonly used in applications such as:

- **Audio & Speech**: analyzing spoken words, music, or animal sounds.
- **Mechanical & Structural Engineering**: vibration monitoring of rotating machinery, bridges, engines, or turbines.
- **Communications & Radar**: understanding how signals vary in time and frequency.
- **Biomedical Engineering**: studying brain signals (EEG), heart signals (ECG), or other physiological data.
- **Seismology / Geophysics**: earthquake analysis — where energy is released at multiple frequencies that evolve over time.
- **Music Technology**: visualizing and editing sounds.

3.4 Wavelets & Multi-Resolution Analysis

The Gábor transform established two key principles for joint time-frequency analysis: *translation*, τ, that slides the short-time window over the time range of the signal and *scaling*, a, that determines the width of the short-time window to capture the frequency components over a fixed time domain. Although the Gábor transform gives a method whereby time and frequency can be simultaneously characterized, there are obvious limitations to the method. Specifically, the method is limited by the time filtering itself. Consider the illustration of them in Fig. (31). The time window filters out the time behavior of the signal in a window centered at τ with width a. Thus, when considering the spectral content of the window, any portion of the signal with a wavelength longer than the window is completely lost. Indeed, since the Heisenberg relationship must hold, the shorter the time-filtering window, the less information there is concerning the frequency content.

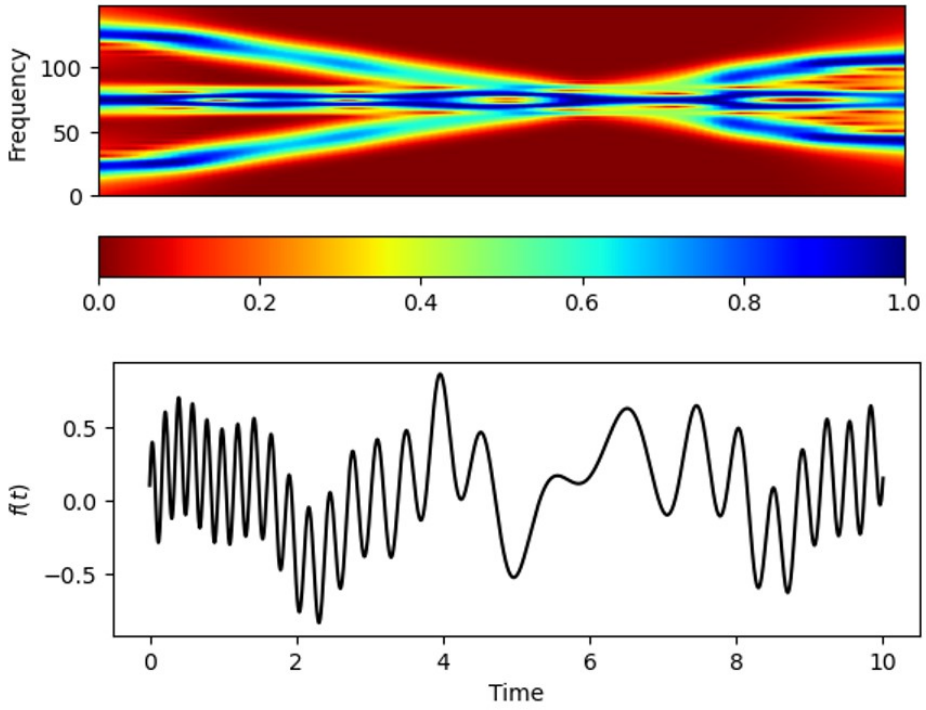

Figure 32: Spectrogram produced from the G*á*bor time-filtering $g(t) = \exp\left(-(t-\tau)^2/a^2\right)$ where $\tau = 4$ and $a = 1$ in the top panel and its associated time-varying signal in the bottom panel.

In contrast, wider windows retain more frequency components, but this comes at the expense of losing the time resolution of the signal. Fig(33) provides a graphical description of the failings of the G*á*bor transform, specifically the trade-offs that occur between time and frequency resolution, and the fact that high accuracy in one of these comes at the expense of resolution in the other parameter. For example, it is known that low-frequency components often last a long period of time, so a high-frequency resolution is required. Whereas high-frequency components often appear as short bursts, invoking the need for a higher time resolution. G*á*bor transform does not have this kind of flexibility to extract the important information when it matters. This is a consequence of a fixed time-filtering window. A simple modification to the G*á*bor transform allows the scaling window, a, to vary to extract improvements in the time resolution. In other words, first, the low-frequency (poor time resolution) components are extracted using a broad scaling window. The scaling window is subsequently shortened in order to extract higher frequencies and better time resolution. By keeping a catalog of the extracting process, both excellent time and frequency resolution of a given signal can be obtained

[5]. This is the fundamental principle of *wavelet theory*. The wavelet transform results in analyzing a signal into different frequencies at different resolutions, known as multiresolution analysis. The term wavelet means little wave and originates from the fact that the scaling window extracts out smaller and smaller pieces of waves from the larger signal. Mathematically, the wavelet transform is given by:

$$\Psi(a,b) = \langle f, \psi_{a,b} \rangle = \int_{-\infty}^{\infty} f(t)\, \psi^*_{a,b}(t)\, \mathrm{d}t \tag{63}$$

where $\psi^*_{a,b}$ is the complex conjugate of $\psi_{a,b}$, which is known as the mother wavelet, which can be stretched (scaled) by a factor of a and shifted (translated) by a factor of b to analyze signals at different frequencies and times. From one "mother wavelet", you generate a whole family of wavelets. The mother wavelet as is given by:

$$\psi_{a,b}(t) = \psi\left(\frac{t-b}{a}\right) \tag{64}$$

where $a \neq 0$ and b are real numbers. The parameter a is the scaling parameter illustrated in Fig. (31) which controls the width where large values of a create stretched wavelet that capture low frequencies, and small values of a create compressed wavelet that capture high frequencies. The parameter b denotes the translation parameter which controls position in time (previously denoted by τ in Fig. (31)).

There is a wide variety of different families (types) of wavelets that can be used based on the values a and b. Each family (type) of wavelets has a different shape, smoothness, and compactness, and is useful for a different purpose. Generally, the Wavelet Transform comes in two different and distinct flavors: the Continuous and the Discrete Wavelet Transform. The main difference between Continuous Wavelet Transform (CWT) and Discrete Wavelet Transform (DWT) lies in their time-frequency representation methods. The CWT provides a continuous representation of signals in both time and frequency domains, while the DWT provides a discrete representation. The DWT is more computationally efficient and is commonly used in practical applications due to its discrete nature and ease of implementation.

3.4.1 Continuous Wavelet Transform

An example of a continuous wavelet is the Mexican hat wavelet, as shown in the bottom plot of Fig. (34,d), which is given by:

$$\psi_{a,b}(t) = \left[1 - \left(\frac{t-b}{a}\right)^2\right] e^{-\frac{1}{2}\left(\frac{t-b}{a}\right)^2} \tag{65}$$

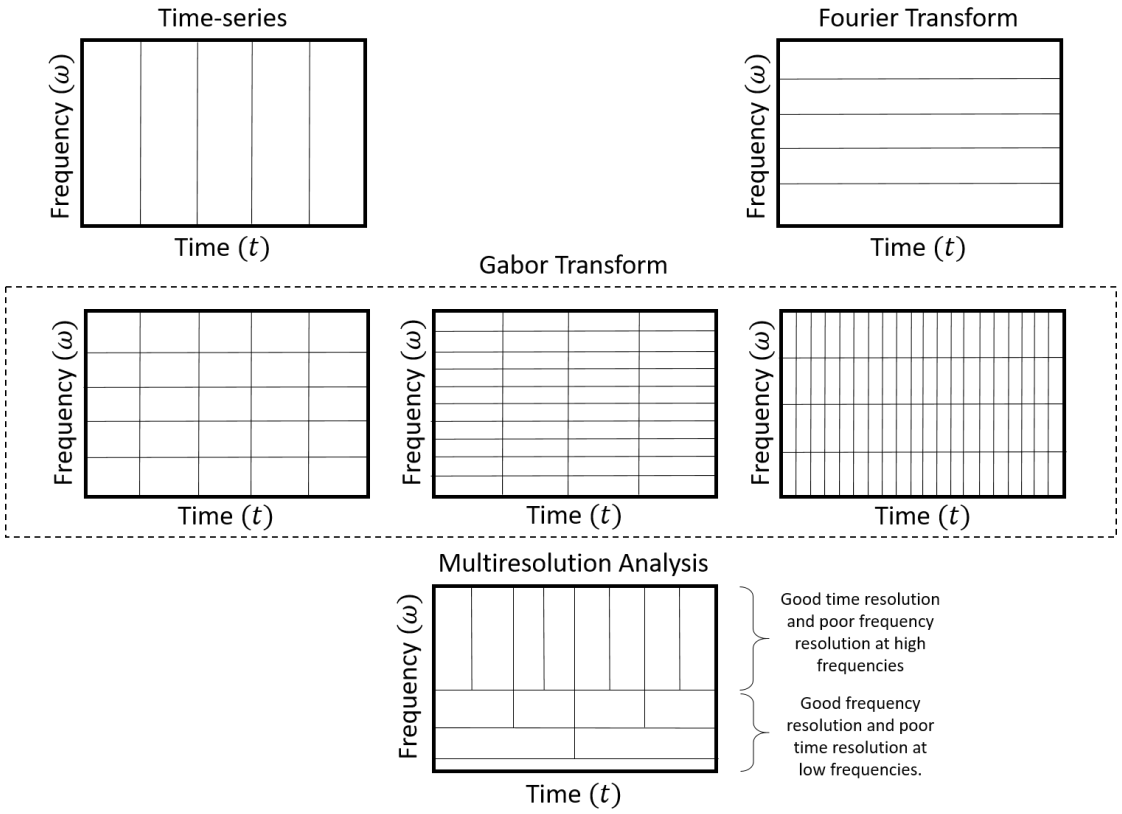

Figure 33: Graphical description of the difference between a time series analysis (top left), Fourier analysis (top right), G*á*bor transform- (middle, left): the size of each box refers to the amount of time/frequency information, (middle, middle): wide windows refer to good accuracy in frequency and less in time, (middle, right): narrow windows refer to a good accuracy in time and less in frequency, and wavelet multiresolution analysis (bottom).

The Mexican hat is, in fact, the second derivative of the Gaussian distribution function $e^{-t^2/2}$: that is, with a unit variance but without the usual $1/\sqrt{2}$ normalization factor. The Mexican hat is the negative of the second derivative of the Gaussian function. All derivatives of the Gaussian function may be employed as a wavelet. Which is the most appropriate one to use depends on the application [17].

3.4.2 Discrete Wavelet Transform

While the CWT is continuous in nature, in the digital realm, we often work with discrete signals. The DWT provides a computationally efficient method to analyze

signals at various resolutions. DWT uses discrete values for the scale and translation factor. The scale factor increases in powers of two, so $a = 1, 2, 4, \cdots$, and the translation factor increases integer values $b = 1, 2, 3\cdot$. The simplest and earliest example of a discrete wavelet is the Haar wavelet, as shown in Fig. (34a,b,c), developed in 1910 [18] and is given by:

$$\psi_{a,b}(t) = \begin{cases} 1 & \text{for} \quad b \leq t < a/2 \\ -1 & \text{for} \quad a/2 \leq t < a \\ 0 & \text{Otherwise} \end{cases} \tag{66}$$

The three Haar wavelets, $\psi_{1,0}(t)$, $\psi_{1/2,0}(t)$, and $\psi_{1/2,1/2}(t)$ are shown in Fig(34,a,b,c), respectively.

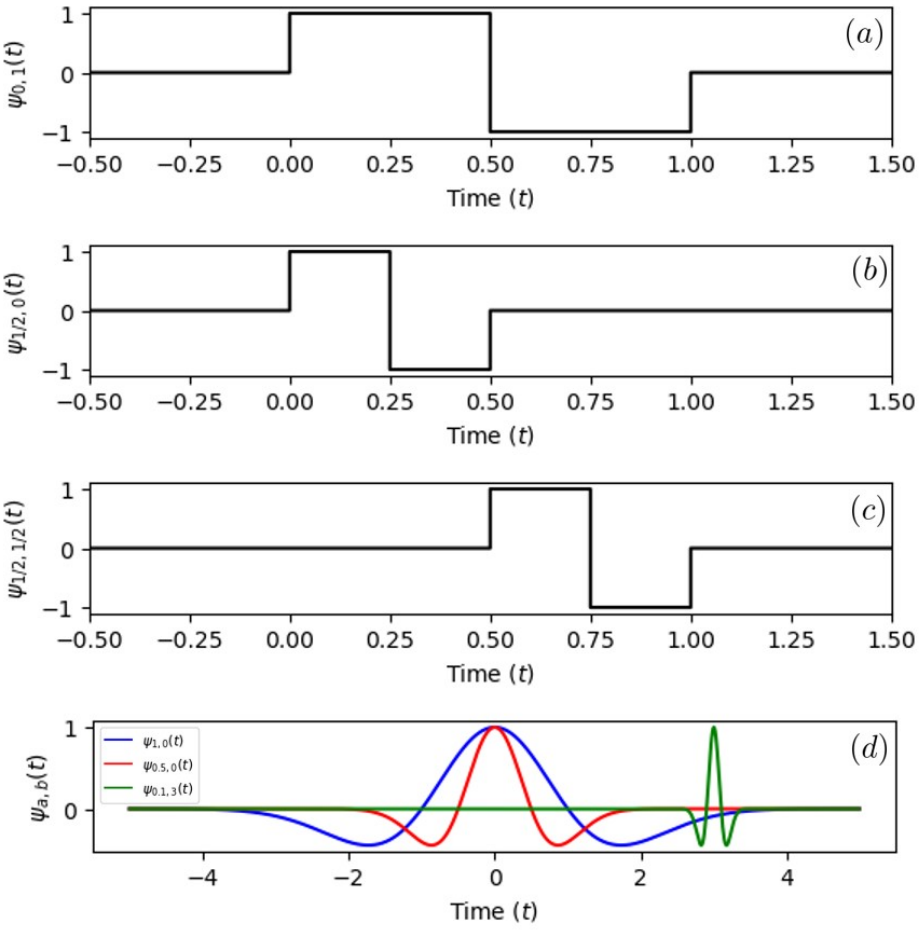

Figure 34: Illustration of two common examples of the mother wavelet. The Haar wavelets: $\psi_{1,0}(t)$ in (a), $\psi_{1/2,0}(t)$ in (b), and $\psi_{1/2,1/2}(t)$ in (c). In (d) is the Mexican hat wavelet, $\psi_{1,0}(t)$, $\psi_{1/2,0}(t)$, and $\psi_{0.1,3}(t)$.

3.5 Scaleogram

As we have seen before, the spectrogram, defined as the absolute value of the short-time Fourier transform, is a very common tool in signal analysis because it provides a distribution of the energy (or intensity or magnitude) of the signal in the time-frequency plane. A similar distribution can be defined in the wavelet case. This leads us to define the wavelet spectrogram, of Scaleogram, as the absolute value of the wavelet transform plotted as a function of time and scale parameter, with the intensity being expressed by a range of colors. Generally speaking, a scaleogram performs better than a spectrogram when the signal is non-stationary, thanks to its multiresolution feature in which the wavelet transform produces high time resolution at high frequencies and high-frequency resolution at low frequencies. In the other way round, the spectrogram has better prediction when the signal is stationary due to its linear resolution. The scaleograms of two time-varying signals (bottom left and bottom right) using continuous (top left and top right) and discrete wavelet (middle left and middle right) functions are shown in Fig. (35). It is clear that these spectrograms show the distribution of the energy of the signal in the time-scale plane, expressed in power per frequency unit, like the spectrogram. However, in contrast to the spectrogram, the energy of the signal is here distributed with different resolutions according to Fig. (33). Scalograms are usually used for non-stationary signals (where frequency content changes quickly and irregularly). They are commonly used in applications such as:

- **Mechanical engineering:** vibration analysis, gear/bearing fault detection.
- **Seismology:** earthquakes and transient signals.
- **Medical signals:** ECG, EEG, detecting sudden bursts or anomalies.
- **Audio processing:** analyzing signals with sudden events (like percussive sounds).

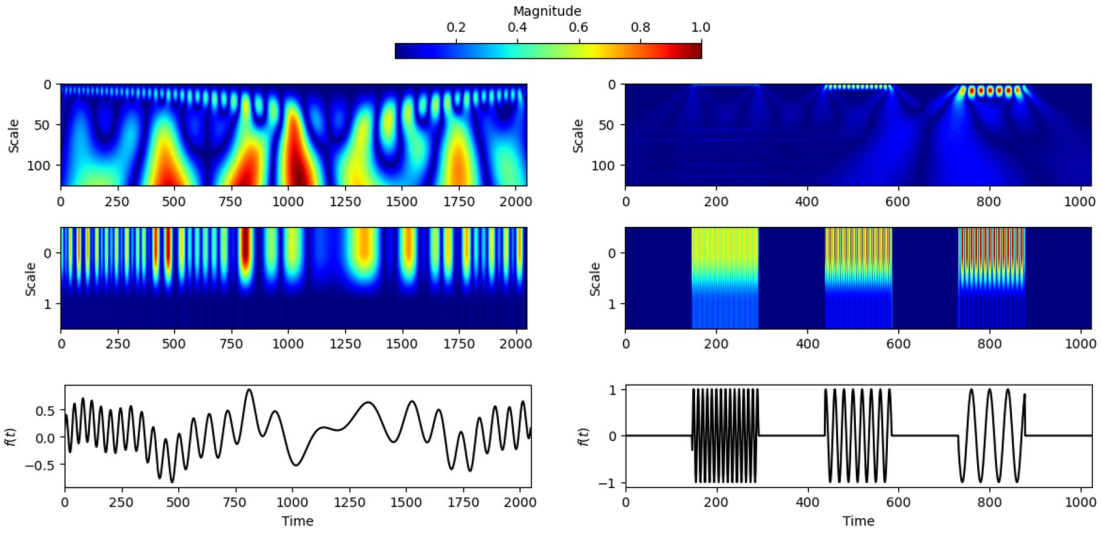

Figure 35: Scaleogram of two time-varying signals (bottom left and bottom right) using a Mexican hat continuous wavelet (top left and top right) and a Haar discrete wavelet (middle left and middle right).