# The Fourier Transform: From Continuous to Discrete and the FFT Algorithm

Your Name

August 10, 2025

## Contents

# 1    Introduction

The Fourier Transform is one of the most powerful mathematical tools in signal processing, physics, and engineering. It allows us to decompose any signal into its constituent frequency components, revealing the "hidden" periodicities within seemingly complex waveforms. This document provides a comprehensive overview of the Fourier Transform, starting from the continuous case, moving to the discrete version, and finally exploring the Fast Fourier Transform (FFT) algorithm.

**Note:** Throughout this document, we use the *unitary (normalized) formulation* of the Fourier Transform, where both the forward and inverse transforms are normalized to ensure energy preservation and symmetry. This means:

- The continuous Fourier Transform and its inverse are normalized by $1/\sqrt{2\pi}$.

- The discrete Fourier Transform (DFT) and its inverse are normalized by $1/\sqrt{N}$.

# 2    The Continuous Fourier Transform

## 2.1    Definition and Intuition

The Continuous Fourier Transform (CFT) of a function $f(t)$ in the unitary (normalized) formulation is defined as:

$$F(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t)e^{-i\omega t}dt \tag{1}$$

where:

- $F(\omega)$ is the frequency domain representation

- $f(t)$ is the time domain signal

- $\omega$ is the angular frequency (radians per second)

- $i$ is the imaginary unit

The inverse Fourier Transform in the unitary formulation is:

$$f(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} F(\omega)e^{i\omega t}d\omega \tag{2}$$

The Fourier Transform has several important properties that make it a powerful tool for analyzing signals. These properties include linearity, time and frequency shifting, and the convolution theorem.

| Property | Description |
|---|---|
| Linearity | $\mathcal{F}\{af(t) + bg(t)\} = aF(\omega) + bG(\omega)$ |
| Time Shifting | $\mathcal{F}\{f(t - t_0)\} = F(\omega)e^{-i\omega t_0}$ |
| Frequency Shifting | $\mathcal{F}\{f(t)e^{i\omega_0 t}\} = F(\omega - \omega_0)$ |
| Convolution Theorem | $\mathcal{F}\{f(t) * g(t)\} = F(\omega) \cdot G(\omega)$ |

Table 1: Key Properties of the Fourier Transform

**Orthonormality of the Continuous Fourier Basis**

The set of functions $\left\{\frac{e^{i\omega t}}{\sqrt{2\pi}}\right\}$ forms an orthonormal basis for square-integrable functions on $\mathbb{R}$. To check orthonormality, we compute the inner product (which for complex functions involves the complex conjugate):

$$\int_{-\infty}^{\infty} \frac{e^{i\omega t}}{\sqrt{2\pi}} \cdot \overline{\frac{e^{i\omega' t}}{\sqrt{2\pi}}} dt = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{i(\omega - \omega')t} dt = \delta(\omega - \omega') \tag{3}$$

where the overline denotes complex conjugation. Orthonormality ensures that the Fourier coefficients are independent and leads to Parseval's theorem (energy conservation).

**Parseval's Theorem (Continuous Case)**

Parseval's theorem states that the total energy of a signal is preserved under the Fourier Transform:

$$\int_{-\infty}^{\infty} |f(t)|^2 dt = \int_{-\infty}^{\infty} |F(\omega)|^2 d\omega \tag{4}$$

This means the energy (or power) of a signal in the time domain is exactly equal to the energy in the frequency domain. This property is a direct consequence of the orthonormality of the Fourier basis and is fundamental in signal processing, physics, and engineering, as it allows analysis and filtering to be performed equivalently in either domain.

## 2.2 Physical Interpretation

The Fourier Transform essentially asks: "How much of each frequency $\omega$ is present in the signal $f(t)$?" The complex exponential $e^{-i\omega t} = \cos(\omega t) - i\sin(\omega t)$ acts as a "probe" that correlates the signal with sinusoidal components at frequency $\omega$.

- **Magnitude** $|F(\omega)|$: represents the amplitude of frequency component $\omega$

- **Phase** $\arg(F(\omega))$: represents the phase shift of that frequency component

Direct computation requires $O(N^2)$ complex multiplications.

## 2.3 Symmetry for Real-Valued Functions

If $f(t)$ is real-valued, its Fourier Transform $F(\omega)$ satisfies the Hermitian symmetry property:

$$F(-\omega) = F^*(\omega) \tag{5}$$

where $F^*$ denotes the complex conjugate. This means the negative frequency components are redundant and determined by the positive frequencies. As a result, the full frequency content of a real signal is captured by the spectrum for $\omega \geq 0$, and the real and imaginary parts of $F(\omega)$ correspond to the amplitudes of cosine and sine components, respectively.

This symmetry is fundamental for interpreting the spectrum of real signals and is analogous to the discrete case. In particular, the real part of $F(\omega)$ for $\omega > 0$ is doubled in the reconstruction of $f(t)$, since the contributions from $+\omega$ and $-\omega$ add together. Thus, the amplitude at each positive frequency is often interpreted as $2 \operatorname{Re}[F(\omega)]$ (or $2|F(\omega)|$ for the magnitude spectrum), reflecting the combined effect of both positive and negative frequencies.

# 3 The Discrete Fourier Transform (DFT)

In practice, we work with discrete, finite-length signals. The Discrete Fourier Transform (DFT) is the discrete analog of the continuous Fourier Transform, designed for sequences of $N$ samples.

For a discrete signal $x[n]$ where $n = 0, 1, 2, \ldots, N - 1$, the DFT is defined as:

$$X[k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x[n] e^{-i2\pi kn/N} \tag{6}$$

$$X[k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x[n] e^{-i2\pi kn/N}$$

where $k = 0, 1, 2, \ldots, N - 1$ are the frequency bin indices.

The inverse DFT (IDFT) is:

$$x[n] = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X[k] e^{i2\pi kn/N} \tag{7}$$

The power spectrum for discrete signals is computed as $P[k] = |X[k]|^2$, where $X[k]$ is the DFT of $x[n]$. This quantity shows the energy content at each frequency bin $k$ and is widely used to analyze the frequency characteristics of digital signals.

**Orthonormality of the Discrete Fourier Basis**

The set of vectors $\left\{ \frac{1}{\sqrt{N}} e^{i2\pi kn/N} \right\}$ for $k = 0, 1, \ldots, N - 1$ forms an orthonormal basis for $\mathbb{C}^N$ under the discrete inner product. The inner product for complex vectors also uses

the complex conjugate:

$$\sum_{n=0}^{N-1} \frac{1}{\sqrt{N}} e^{i2\pi kn/N} \cdot \overline{\frac{1}{\sqrt{N}} e^{i2\pi k'n/N}} = \frac{1}{N} \sum_{n=0}^{N-1} e^{i2\pi(k-k')n/N} = \delta_{k,k'} \tag{8}$$

where the overline denotes complex conjugation, just as in the continuous case. This is the complex analog of the dot product. This property guarantees the independence of frequency components and underpins Parseval's theorem in the discrete case.

**Parseval's Theorem (Discrete Case)**

In the discrete case, Parseval's theorem states:

$$\sum_{n=0}^{N-1} |x[n]|^2 = \sum_{k=0}^{N-1} |X[k]|^2 \tag{9}$$

That is, the sum of the squared magnitudes of the signal samples equals the sum of the squared magnitudes of its DFT coefficients. This is crucial for digital signal processing, as it guarantees that no energy is lost or gained when transforming between time and frequency domains, and enables energy-based analysis and filtering in either domain.

## 3.1 Frequency Interpretation

The DFT computes the spectrum at $N$ equally spaced frequency points:

$$\omega_k = \frac{2\pi k}{N}, \quad k = 0, 1, \ldots, N-1 \tag{10}$$

Even though $k$ only ranges from 0 to $N-1$, these frequency indices actually wrap around the unit circle in the complex plane. This means the DFT is periodic in frequency with period $N$, and the upper half of the frequency bins (for $k > N/2$) can be interpreted as negative frequencies. Specifically, the frequency bin $k$ corresponds to a frequency of $\omega_k = 2\pi k/N$, but for $k > N/2$, this is equivalent to $\omega_k - 2\pi = 2\pi(k - N)/N$, which is a negative frequency. As a result, the DFT spectrum is often visualized by shifting the zero-frequency component to the center (using, for example, the 'fftshift' operation in software), so that negative frequencies appear on the left and positive frequencies on the right. This makes the symmetry and interpretation of the spectrum more apparent, especially for real-valued signals.

If the sampling rate is $f_s$, then the actual frequencies are:

$$f_k = \frac{k f_s}{N}, \quad k = 0, 1, \ldots, N-1 \tag{11}$$

## 3.2   Symmetry for Real Signals

When the input signal $x[n]$ is real-valued, the DFT exhibits a special symmetry known as Hermitian symmetry:

$$X[k] = X^*[N - k] \tag{12}$$

where $X^*$ denotes the complex conjugate and $N$ is the total number of points. This means the spectrum is redundant for negative frequencies, and the information for $k$ and $N - k$ is related.

As a result, the DFT of a real signal can be fully described by the coefficients for $k = 0$ to $N/2$ (for even $N$). The real part of $X[k]$ gives the amplitude of the cosine component at frequency $k$, and the imaginary part gives the amplitude of the sine component. The magnitude spectrum is often plotted as $2|X[k]|$ for $k > 0$ to account for the energy shared between positive and negative frequencies.

This symmetry is important for interpreting the frequency content of real-world signals (such as audio, images, and measurements), and is exploited in many FFT implementations to save computation and storage.

## 3.3   Key Differences from CFT

- **Periodicity**: Both $x[n]$ and $X[k]$ are periodic with period $N$

- **Finite**: We only consider $N$ samples in both domains

- **Sampling**: The frequency domain is also sampled (quantized)

## 3.4   Windowing

When applying the DFT to finite-length signals, edge effects can cause spectral leakage. Windowing functions (Hamming, Hanning, Blackman) are applied to mitigate these effects:

$$x_w[n] = x[n] \cdot w[n] \tag{13}$$

where $w[n]$ is the window function.

## 3.5   Zero-Padding

Zero-padding increases frequency resolution and can improve the appearance of the spectrum:

$$x_{padded}[n] = \begin{cases} x[n] & \text{if } 0 \leq n < N \\ 0 & \text{if } N \leq n < M \end{cases} \tag{14}$$

where $M > N$ is the new length.

# 4    Fourier Transform for Parametrized Curves

For a parametrized curve $[x(t), y(t)]$, there are two common approaches to Fourier analysis:

**Component-wise:** Apply the Fourier Transform separately to $x(t)$ and $y(t)$.

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-i\omega t}dt \qquad Y(\omega) = \int_{-\infty}^{\infty} y(t)e^{-i\omega t}dt \qquad (15)$$

**Complex signal:** Combine the components into a single complex signal $z(t) = x(t) + iy(t)$ and analyze $z(t)$.

$$Z(\omega) = \int_{-\infty}^{\infty} [x(t) + iy(t)]e^{-i\omega t}dt \qquad (16)$$

Both approaches are useful and reveal different aspects of the curve's frequency content. Treating $x(t)$ and $y(t)$ separately is simple and useful when the components are independent. The complex approach is powerful for capturing phase relationships and correlations, and is often used in shape and motion analysis. Comparing both the individual and complex spectra can provide deeper insight into the structure and dynamics of the curve. For either approach, the power spectrum is given by the squared magnitude of the transform:

- **Component-wise:** $|X(\omega)|^2$ and $|Y(\omega)|^2$ (can be summed for total power).

- **Complex signal:** $|Z(\omega)|^2$ (includes cross-terms and phase relationships).

## 4.1    Applications

Fourier analysis of parametrized curves is useful in:

- **Shape Analysis**: Decomposing shapes into frequency components for comparison or recognition.

- **Motion Analysis**: Analyzing trajectories in terms of their frequency content.

- **Signal Processing**: Studying periodic patterns in two-dimensional data.

## 4.2    Examples

- **Epitrochoid:** An epitrochoid is a type of roulette curve traced by a point attached to a circle of radius $r$ rolling around the outside of a fixed circle of radius $R$. The parameter $d$ controls the distance from the center of the rolling circle to the tracing point. These curves can create intricate, spirograph-like patterns and are useful

for demonstrating how complex periodic shapes can be decomposed into frequency components.

$$x(t) = (R+r)\cos t + d\cos\left(\frac{R+r}{r}t\right) \tag{17}$$

$$y(t) = (R+r)\sin t + d\sin\left(\frac{R+r}{r}t\right) \tag{18}$$

(e.g., $R = 5$, $r = 1$, $d = 2$)

- **Rose curve:** The rose curve is a polar curve with petal-like shapes, defined by a sum of cosines with different frequencies. The number and shape of the petals depend on the coefficients and frequencies used. Rose curves are classic examples in Fourier analysis because their symmetry and periodicity make their frequency content easy to interpret.

$$r(t) = 3\cos(3t) + \cos(9t) + 0.5\cos(15t) \tag{19}$$
$$x(t) = r(t)\cos t \tag{20}$$
$$y(t) = r(t)\sin t \tag{21}$$

- **Lissajous curve:** Lissajous curves are generated by combining two sinusoidal oscillations in perpendicular directions, often with different frequencies and a phase shift. They are used to illustrate the relationship between frequency, phase, and shape, and are common in signal analysis and physics demonstrations.

$$x(t) = 4\sin(3t) + 1.5\sin(6t) + 0.8\sin(9t) \tag{22}$$
$$y(t) = 3\sin(5t + \phi) + 1.2\sin(10t) + 0.6\sin(15t) \tag{23}$$

(with $\phi = \pi/4$)

- **Spirograph-like:** This example combines several sine and cosine terms with different amplitudes and frequencies to create a complex, closed curve reminiscent of patterns made by a spirograph toy. Such curves demonstrate how adding harmonics (higher-frequency components) affects the overall shape, and how Fourier series can represent intricate periodic patterns.

$$x(t) = 3\cos t + 2\cos(2t) + 1\cos(4t) + 0.5\cos(7t) \tag{24}$$
$$y(t) = 3\sin t + 2\sin(2t) + 1\sin(4t) + 0.5\sin(7t) \tag{25}$$

# 5 The Two-Dimensional Fourier Transform

The two-dimensional (2D) Fourier Transform is a natural extension of the 1D case and is widely used in fields such as image processing, optics, and data analysis. It allows us to analyze the frequency content of functions defined over a plane, such as images or spatial data.

## 5.1 Definition

Given a function $f(x, y)$ defined over $\mathbb{R}^2$, the 2D continuous Fourier Transform is defined as:

$$F(u, v) = \iint_{-\infty}^{\infty} f(x, y)e^{-i(ux+vy)}dxdy \tag{26}$$

where $(u, v)$ are the frequency variables corresponding to the $x$ and $y$ directions.

The inverse 2D Fourier Transform is:

$$f(x, y) = \frac{1}{(2\pi)^2} \iint_{-\infty}^{\infty} F(u, v)e^{i(ux+vy)}dudv \tag{27}$$

For discrete data (such as digital images), the 2D Discrete Fourier Transform (DFT) is:

$$F[k, l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m, n]e^{-i2\pi(\frac{km}{M} + \frac{ln}{N})} \tag{28}$$

where $f[m, n]$ is the value at pixel $(m, n)$ and $F[k, l]$ is the frequency component at $(k, l)$.

## 5.2 Properties and Interpretation

The 2D Fourier Transform decomposes a function into its spatial frequency components. Each point in the frequency domain represents a particular combination of horizontal and vertical frequencies. The magnitude $|F(u, v)|$ indicates the strength of that frequency, while the phase encodes spatial alignment.

## 5.3 Applications

- **Image Processing**: Filtering, denoising, edge detection, and image compression (e.g., JPEG uses the 2D DCT, a close relative of the 2D FT).

- **Pattern Recognition**: Identifying repeating structures or textures in images.

- **Optics**: Analyzing diffraction patterns and lens systems.

- **Data Analysis**: Studying spatial correlations in 2D datasets (e.g., geophysics, astronomy).

## 5.4 Example: Image Filtering

A common use of the 2D Fourier Transform is to filter images. For example, a low-pass filter can be applied in the frequency domain to remove high-frequency noise, resulting in a smoother image. Conversely, a high-pass filter can enhance edges and fine details.

**Steps:**

1. Compute the 2D DFT of the image.

2. Modify the frequency components (e.g., set high frequencies to zero for low-pass filtering).

3. Compute the inverse 2D DFT to obtain the filtered image.

The 2D Fourier Transform is a powerful tool for understanding and manipulating spatial data, and forms the mathematical foundation for many modern imaging techniques.

# 6 The Fast Fourier Transform (FFT)

The Fast Fourier Transform (FFT) is a family of algorithms for efficiently computing the Discrete Fourier Transform (DFT) and its inverse. While the DFT provides the mathematical foundation for frequency analysis of discrete signals, the FFT makes it practical to compute the DFT for large datasets by reducing the computational complexity from $O(N^2)$ to $O(N \log N)$. Understanding the FFT is not required for grasping the core ideas of Fourier analysis, but it is essential for real-world applications.

## 6.1 The Computational Challenge

Computing the DFT directly requires $N^2$ complex multiplications and additions. For large $N$ (e.g., $N = 1024$ or $N = 4096$), this becomes computationally expensive. The FFT algorithm reduces this complexity from $O(N^2)$ to $O(N \log N)$.

## 6.2 The Cooley-Tukey Algorithm

The most common FFT algorithm is the Cooley-Tukey algorithm, which uses a divide-and-conquer approach. We'll focus on the radix-2 decimation-in-time version, which requires $N$ to be a power of 2.

### 6.2.1 Key Insight: Divide and Conquer

The main idea is to recursively break down the DFT of length $N$ into two DFTs of length $N/2$:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \tag{29}$$

$$= \sum_{n \text{ even}} x[n] W_N^{kn} + \sum_{n \text{ odd}} x[n] W_N^{kn} \tag{30}$$

Let $n = 2m$ for even indices and $n = 2m + 1$ for odd indices:

$$X[k] = \sum_{m=0}^{N/2-1} x[2m]W_N^{2km} + \sum_{m=0}^{N/2-1} x[2m+1]W_N^{k(2m+1)} \tag{31}$$

$$= \sum_{m=0}^{N/2-1} x[2m]W_{N/2}^{km} + W_N^k \sum_{m=0}^{N/2-1} x[2m+1]W_{N/2}^{km} \tag{32}$$

where we used the identity $W_N^{2km} = W_{N/2}^{km}$.
This gives us:

$$X[k] = E[k] + W_N^k \cdot O[k] \tag{33}$$

where:

- $E[k]$ is the DFT of the even-indexed samples

- $O[k]$ is the DFT of the odd-indexed samples

- $W_N^k = e^{-i2\pi k/N}$ is the "twiddle factor"

### 6.2.2 Periodicity and Symmetry

Since $E[k]$ and $O[k]$ are periodic with period $N/2$, we have:

$$X[k] = E[k] + W_N^k \cdot O[k], \quad k = 0, 1, \ldots, N/2 - 1 \tag{34}$$

$$X[k + N/2] = E[k] - W_N^k \cdot O[k], \quad k = 0, 1, \ldots, N/2 - 1 \tag{35}$$

This symmetry allows us to compute all $N$ output points using only $N/2$ evaluations of each sub-DFT.

## 6.3 FFT Algorithm Pseudocode

extbfPseudocode: Cooley-Tukey FFT (Radix-2, Decimation-in-Time)

```
Input: x[0..N-1] where N = 2^m
Output: X[0..N-1] (DFT of x)

if N = 1:
    return x[0]
else:
    x_even = [x[0], x[2], ..., x[N-2]]
    x_odd  = [x[1], x[3], ..., x[N-1]]
    E = FFT(x_even)
    O = FFT(x_odd)
    for k = 0 to N/2-1:
        t = exp(-i*2*pi*k/N) * O[k]
```

```
        X[k] = E[k] + t
        X[k+N/2] = E[k] - t
    return X
```

## 6.4   Computational Complexity

The FFT reduces the computational complexity from $O(N^2)$ to $O(N \log N)$:

- **Levels**: $\log_2 N$ levels of recursion

- **Operations per level**: $N$ complex multiplications and additions

- **Total**: $N \log_2 N$ operations

For $N = 1024$:

- Direct DFT: $1024^2 = 1,048,576$ operations

- FFT: $1024 \times 10 = 10,240$ operations

- Speedup: $\sim 100\times$

## 6.5   Bit-Reversal and In-Place Implementation

The recursive formulation can be converted to an iterative, in-place algorithm using bit-reversal ordering. The input array is first rearranged so that element at position $n$ is moved to position obtained by reversing the binary representation of $n$.

For example, with $N = 8$:

| Decimal | Binary | Bit-Reversed | New Position |
|---------|--------|--------------|--------------|
| 0 | 000 | 000 | 0 |
| 1 | 001 | 100 | 4 |
| 2 | 010 | 010 | 2 |
| 3 | 011 | 110 | 6 |
| 4 | 100 | 001 | 1 |
| 5 | 101 | 101 | 5 |
| 6 | 110 | 011 | 3 |
| 7 | 111 | 111 | 7 |

## 6.6   The 2D Fast Fourier Transform

The Fast Fourier Transform can also be extended to two dimensions, making it possible to efficiently compute the 2D DFT of images and other 2D data. The 2D FFT works by applying the 1D FFT algorithm first along one axis (e.g., rows) and then along the other axis (e.g., columns). This reduces the computational complexity from $O(N^2 M^2)$ to $O(NM \log(NM))$ for an $N \times M$ array.

Mathematically, for a 2D array $f[m,n]$ of size $M \times N$, the 2D DFT is:

$$F[k,l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n] e^{-i2\pi(\frac{km}{M} + \frac{ln}{N})} \tag{36}$$

The 2D FFT computes this efficiently by:

1. Applying the 1D FFT to each row:

$$f'[m,l] = \sum_{n=0}^{N-1} f[m,n] e^{-i2\pi \frac{ln}{N}} \tag{37}$$

2. Then applying the 1D FFT to each column of the result:

$$F[k,l] = \sum_{m=0}^{M-1} f'[m,l] e^{-i2\pi \frac{km}{M}} \tag{38}$$

The same process can be used for the inverse 2D FFT, applying the inverse 1D FFT along columns and then rows.

The 2D FFT is widely used in image processing for filtering, compression, and feature extraction, as well as in scientific computing and data analysis involving spatial data.

# 7 Conclusion

The Fourier Transform provides a fundamental bridge between time and frequency domains, enabling us to analyze and manipulate signals in ways that would be impossible in a single domain. The progression from the continuous Fourier Transform to the discrete version, and finally to the computationally efficient FFT algorithm, demonstrates how mathematical theory translates into practical computational tools.

The FFT's $O(N \log N)$ complexity has made real-time spectral analysis possible in countless applications, from digital audio processing to medical imaging. Understanding these concepts provides the foundation for advanced signal processing techniques and helps in choosing appropriate tools for specific applications.

# 8 References

- Cooley, J. W., & Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. Mathematics of Computation, 19(90), 297-301.

- Oppenheim, A. V., & Schafer, R. W. (2009). Discrete-time signal processing. Pearson.

- Brigham, E. O. (1988). The fast Fourier transform and its applications. Prentice Hall.

- Gonzalez, R. C., & Woods, R. E. (2018). Digital Image Processing (4th Edition). Pearson.

- Bracewell, R. N. (2000). The Fourier Transform and Its Applications (3rd Edition). McGraw-Hill.

- Harris, F. J. (1978). On the use of windows for harmonic analysis with the discrete Fourier transform. Proceedings of the IEEE, 66(1), 51-83.

- Smith, S. W. (1997). The Scientist and Engineer's Guide to Digital Signal Processing. California Technical Publishing.