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

Your Name

August 6, 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.

2 The Continuous Fourier Transform

2.1 Definition and Intuition

The Continuous Fourier Transform (CFT) of a function $f(t)$ is defined as:

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

where:

- $F(\omega)$ is the frequency domain representation
- $f(t)$ is the time domain signal
- ω is the angular frequency (radians per second)
- i is the imaginary unit

The inverse Fourier Transform recovers the original signal:

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

2.2 Physical Interpretation

The Fourier Transform essentially asks: "How much of each frequency ω 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 ω.

When we compute $F(\omega)$, we obtain:

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

The squared magnitude $|F(\omega)|^2$ is called the **power spectrum** of the signal. It describes how the energy of the signal is distributed among different frequencies. Peaks in the power spectrum indicate dominant frequencies present in the original signal. The power spectrum is a fundamental tool in signal analysis, audio and image processing, and many areas of physics and engineering.

2.3 Key Properties

2.3.1 Linearity

If $\mathcal{F}\{f(t)\} = F(\omega)$ and $\mathcal{F}\{g(t)\} = G(\omega)$, then:

$$\mathcal{F}\{af(t) + bg(t)\} = aF(\omega) + bG(\omega) \tag{3}$$

2.3.2 Time Shifting

$$\mathcal{F}\{f(t - t_0)\} = F(\omega)e^{-i\omega t_0} \tag{4}$$

2.3.3 Frequency Shifting

$$\mathcal{F}\{f(t)e^{i\omega_0 t}\} = F(\omega - \omega_0) \tag{5}$$

2.3.4 Convolution Theorem

$$\mathcal{F}\{f(t) * g(t)\} = F(\omega) \cdot G(\omega) \tag{6}$$

where $*$ denotes convolution.

3 The Discrete Fourier Transform (DFT)

3.1 Motivation and Definition

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:

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

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

The inverse DFT (IDFT) is:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k]e^{i2\pi kn/N} \tag{8}$$

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.

3.2 Frequency Interpretation

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

$$\omega_k = \frac{2\pi k}{N}, \quad k = 0, 1, \ldots, N-1 \tag{9}$$

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

$$f_k = \frac{k f_s}{N}, \quad k = 0, 1, \ldots, N-1 \tag{10}$$

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:

$$x_w[n] = x[n] \cdot w[n] \tag{11}$$

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

3.5 Zero-Padding

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

$$x_{padded}[n] = \begin{cases} x[n] & \text{if } 0 \leq n < N \\ 0 & \text{if } N \leq n < M \end{cases} \tag{12}$$

where $M > N$ is the new length.

3.6 Matrix Representation

The DFT can be written as a matrix multiplication:

$$\mathbf{X} = \mathbf{W}_N \mathbf{x} \tag{13}$$

where $\mathbf{W}_N$ is the $N \times N$ DFT matrix with elements:

$$[\mathbf{W}_N]_{k,n} = e^{-i2\pi kn/N} \tag{14}$$

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

4 Fourier Transform for Parametrized Curves

4.1 Introduction

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

Both approaches are useful and reveal different aspects of the curve's frequency content.

4.2 Definition

Component-wise:

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

$$Y(\omega) = \int_{-\infty}^{\infty} y(t)e^{-i\omega t}dt \tag{16}$$

Complex signal:

$$Z(\omega) = \int_{-\infty}^{\infty} [x(t) + iy(t)]e^{-i\omega t}dt \tag{17}$$

4.3 Reconstruction

Component-wise:

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\omega)e^{i\omega t}d\omega \tag{18}$$

$$y(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} Y(\omega)e^{i\omega t}d\omega \tag{19}$$

Complex signal:

$$z(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} Z(\omega)e^{i\omega t}d\omega \tag{20}$$

where $z(t) = x(t) + iy(t)$.

4.4 Power Spectrum

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).

4.5 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.

4.6 Examples

Circle: $x(t) = \cos(t)$, $y(t) = \sin(t)$. As a complex signal, $z(t) = e^{it}$, whose Fourier Transform has a single frequency component.

Lissajous Curve:

$$x(t) = \cos(2t) \tag{21}$$
$$y(t) = \sin(3t) \tag{22}$$

Component-wise, the spectra have peaks at frequencies 2 and 3. As a complex signal, $z(t) = \cos(2t) + i\sin(3t)$, the spectrum reflects both frequencies and their phase relationship.

4.7 Choosing an Approach

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.

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:

$$F(u, v) = \iint_{-\infty}^{\infty} f(x, y) e^{-i(ux+vy)} dx dy \tag{23}$$

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

The inverse 2D Fourier Transform is:

$$f(x,y) = \frac{1}{(2\pi)^2} \iint_{-\infty}^{\infty} F(u,v) e^{i(ux+vy)} du dv \tag{24}$$

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

$$F[k,l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n] e^{-i2\pi(\frac{km}{M}+\frac{ln}{N})} \tag{25}$$

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 $N/2$:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \tag{26}$$

$$= \sum_{n \text{ even}} x[n] W_N^{kn} + \sum_{n \text{ odd}} x[n] W_N^{kn} \tag{27}$$

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

$$X[k] = \sum_{m=0}^{N/2-1} x[2m] W_N^{2km} + \sum_{m=0}^{N/2-1} x[2m+1] W_N^{k(2m+1)} \tag{28}$$

$$= \sum_{m=0}^{N/2-1} x[2m] W_{N/2}^{km} + W_N^k \sum_{m=0}^{N/2-1} x[2m+1] W_{N/2}^{km} \tag{29}$$

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

$$X[k] = E[k] + W_N^k \cdot O[k] \tag{30}$$

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:

$$X[k] = E[k] + W_N^k \cdot O[k], \quad k = 0, 1, \ldots, N/2 - 1 \tag{31}$$

$$X[k + N/2] = E[k] - W_N^k \cdot O[k], \quad k = 0, 1, \ldots, N/2 - 1 \tag{32}$$

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

6.3 FFT Algorithm Pseudocode

Algorithm 1 Cooley-Tukey FFT (Radix-2, Decimation-in-Time)

Require: $x[0..N-1]$ where $N = 2^m$
Ensure: $X[0..N-1]$ (DFT of x)
if $N = 1$ **then**
  return $x[0]$
end if
$x_{even} \leftarrow [x[0], x[2], x[4], \ldots, x[N-2]]$
$x_{odd} \leftarrow [x[1], x[3], x[5], \ldots, x[N-1]]$
$E \leftarrow \text{FFT}(x_{even})$ {Recursive call}
$O \leftarrow \text{FFT}(x_{odd})$ {Recursive call}
for $k = 0$ to $N/2 - 1$ **do**
  $t \leftarrow e^{-i2\pi k/N} \cdot O[k]$ {Twiddle factor multiplication}
  $X[k] \leftarrow E[k] + t$
  $X[k + N/2] \leftarrow E[k] - t$
end for
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 $O(N^2M^2)$ 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:

$$F[k, l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m, n] e^{-i2\pi(\frac{km}{M} + \frac{ln}{N})} \tag{33}$$

The 2D FFT computes this efficiently by:

1. Applying the 1D FFT to each row:

$$f'[m, l] = \sum_{n=0}^{N-1} f[m, n] e^{-i2\pi \frac{ln}{N}} \tag{34}$$

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

$$F[k, l] = \sum_{m=0}^{M-1} f'[m, l] e^{-i2\pi \frac{km}{M}} \tag{35}$$

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.