Fourier Transforms!

PHYS 250 (Autumn 2024) – Lecture 11

David Miller

Department of Physics and the Enrico Fermi Institute
University of Chicago

November 5, 2024

Outline

1. *Plan going forward*
   - Data analysis tools
2. *The Square Wave*
   - Square Wave
3. *Extension to the Fourier Transform*
   - Euler's Formula
   - Fourier Transforms

Moving towards physics data analysis

As we discussed last time, I would like to take the direction of this quarter more towards practical physics data analysis and algorithms. We will start with **Fourier Transforms** and **Neural Networks** and then analyze data from the **CMB, LIGO, and/or the Large Hadron Collider**.

Fourier Analysis and Neural Networks

- **Fourier Series and Analysis:**
  - Discussing the basics of Fourier Series
  - Evaluate, computationally, the coefficients of a simple series for both a square and sawtooth wave
  - Extend discussion to the **Fast Fourier Transform**
- **Neural Networks:**
  - Training computers to discover, identify, and analyze patterns in data
  - Modeling perspective on what a neural network achieves
  - Structure and function of a neuron
  - Mathematical properties of a neural network

This will be a mixture of Python Notebooks and Lecture Slides

Outline

Square Wave

Let's start with the Fourier Series for the **square wave** function.

As you may have heard in a variety of contexts (e.g. PHYS 133!) we can decompose any periodic function or periodic signal into the weighted sum of a (possibly infinite) set of simple oscillating functions, namely sines and cosines (or, equivalently, complex exponentials).

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

This is possible because the trigonometric functions for a **set of complete, orthogonal basis vectors** that span the space.

Now open up the `Fourier-Series.ipynb` jupyter notebook and we will discuss this more deeply!

Fourier series for a square wave

We may determine the coefficients of a sine and cosine expansion to be:

$$a_n = \frac{2}{n\pi} \sin\left(n\omega_0 \frac{\pi}{2}\right) \tag{2}$$

which yields a discrete function:

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos(nx) \tag{3}$$

Now open up the `Fourier-Transforms-Analysis.ipynb` jupyter notebook so that we can look at this in more detail!

Outline

1. *Plan going forward*
   - Data analysis tools
2. *The Square Wave*
   - Square Wave
3. *Extension to the Fourier Transform*
   - Euler's Formula
   - Fourier Transforms

Euler's Formula

We can generalize this by making use of Euler's Formula.

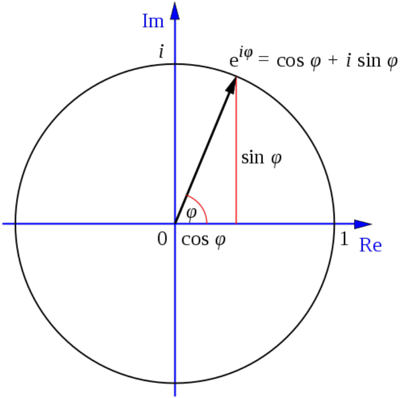
Euler's formula states that for any real number ϕ:

$$e^{i\phi} = \cos(\phi) + i\sin(\phi) \tag{4}$$

When $\phi = \pi$, Euler's formula evaluates to

$$e^{i\pi} + 1 = 0, \tag{5}$$

which is known as Euler's identity.



The implication is that it is possible to recover the amplitude of each wave in a Fourier series using an integral, which has many useful properties (in particular, that it's then continuous).

Fourier Transforms (I)

I will use the following definitions for the Fourier transform $\hat{f}(\xi)$ of a function $f(x)$, where x typically represents either a **spatial or time domain**, and ξ typically represents a corresponding inverse notion of **spatial or time frequency**.

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i \xi x} dx \tag{6}$$

The **inverse transform** is then obtained via

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi) e^{2\pi i \xi x} d\xi \tag{7}$$

In the case of spatial coordinates, x denotes length and ξ denotes inverse wavelength: $\xi = \frac{1}{\lambda}$. In the time domain, x denotes time and ξ denotes frequency. In the case that $x = t$ is in seconds, but ξ is **angular** frequency ω then a factor of 2π appears to get the normalization correct.

Fourier Transforms (II)

In the case of spatial coordinates, x denotes length and ξ denotes inverse wavelength: $\xi = \frac{1}{\lambda}$. In the time domain, x denotes time and ξ denotes frequency. In the case that $x = t$ is in seconds, but ξ is **angular** frequency ω then a factor of 2π appears to get the normalization correct.

$$\hat{f}(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt \tag{8}$$

$$f(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(\omega) e^{i\omega t} d\omega \tag{9}$$

Since $\omega = 2\pi\xi = \frac{2\pi}{\lambda}$.

The $\frac{1}{\sqrt{2\pi}}$ factor in both these integrals is a common normalization in quantum mechanics but maybe not in engineering where only a single $\frac{1}{2\pi}$ factor is often used.

Discrete Fourier Transforms (I)

If $\hat{f}(\omega)$ or $f(t)$ are known analytically or numerically, the Fourier transform integrals can be evaluated using the integration techniques studied earlier.

Discrete Fourier Transforms (I)

If $\hat{f}(\omega)$ or $f(t)$ are known analytically or numerically, the Fourier transform integrals can be evaluated using the integration techniques studied earlier.

In practice, the signal $f(t)$ is measured, or **sampled** at just a finite number N of times t, and these are what we must use to approximate the transform.

Discrete Fourier Transforms (I)

If $\hat{f}(\omega)$ or $f(t)$ are known analytically or numerically, the Fourier transform integrals can be evaluated using the integration techniques studied earlier.

In practice, the signal $f(t)$ is measured, or **sampled** at just a finite number N of times t, and these are what we must use to approximate the transform.

The resultant **discrete Fourier transform (DFT)** is an approximation both because the signal is not known for all times and because we integrate numerically.

Discrete Fourier Transforms (I)

If $\hat{f}(\omega)$ or $f(t)$ are known analytically or numerically, the Fourier transform integrals can be evaluated using the integration techniques studied earlier.

In practice, the signal $f(t)$ is measured, or **sampled** at just a finite number N of times t, and these are what we must use to approximate the transform.

The resultant **discrete Fourier transform (DFT)** is an approximation both because the signal is not known for all times and because we integrate numerically.

Once we have a discrete set of transforms, they can be used to reconstruct the signal for any value of the time.

Discrete Fourier Transforms (I)

If $\hat{f}(\omega)$ or $f(t)$ are known analytically or numerically, the Fourier transform integrals can be evaluated using the integration techniques studied earlier.

In practice, the signal $f(t)$ is measured, or **sampled** at just a finite number N of times t, and these are what we must use to approximate the transform.

The resultant **discrete Fourier transform (DFT)** is an approximation both because the signal is not known for all times and because we integrate numerically.

Once we have a discrete set of transforms, they can be used to reconstruct the signal for any value of the time.

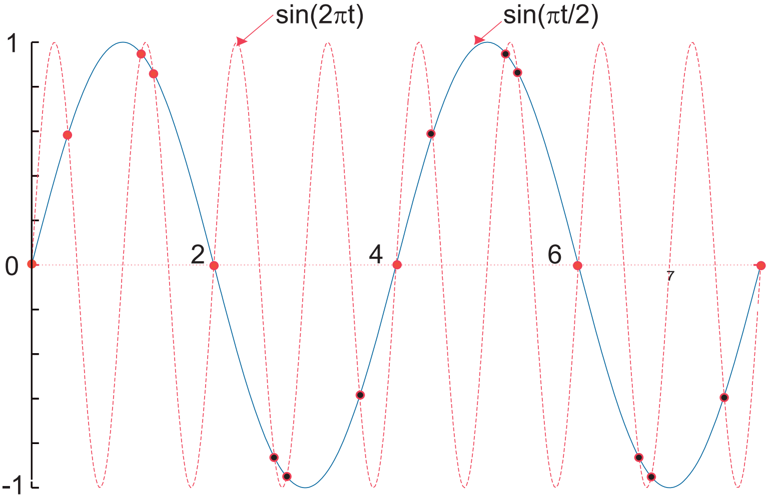
In this way the **DFT can be thought of as a technique for interpolating, compressing, and extrapolating data**.

Discussion

Do you see any issues with this "sampling"?

Discussion

Do you see any issues with this "sampling"?



Discrete Fourier Transforms (II)

The DFT algorithm results from evaluating the integral not from 1 to $+1$ but rather from time 0 to time T over which the signal is measured, and from approximating the integration of the integral by computing a discrete sum:

$$\hat{f}(\omega_n) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t) e^{-i\omega_n t} dt \tag{10}$$

$$\simeq \frac{1}{\sqrt{2\pi}} \int_{0}^{T} f(t) e^{-i\omega_n t} dt \tag{11}$$

$$\simeq \frac{1}{\sqrt{2\pi}} \sum_{k=1}^{N} h f(t_k) e^{-i\omega_n t_k} \qquad (h \equiv \text{stepsize}) \tag{12}$$

$$\simeq \frac{h}{\sqrt{2\pi}} \sum_{k=1}^{N} f_k e^{-2\pi i k n/N} \tag{13}$$

$$\hat{f}_n \equiv \frac{\hat{f}(\omega_n)}{h} = \frac{1}{\sqrt{2\pi}} \sum_{k=1}^{N} f_k e^{-2\pi i k n/N} \tag{14}$$

Discrete Fourier Transforms (III)

We then need the inverse as well, which we can obtain with $d\omega \to 2\pi/Nh$ we invert the $\hat{f}_n$

$$f_k = \frac{1}{\sqrt{2\pi}} \sum_{n=1}^{N} \frac{2\pi}{Nh} \hat{f}_n e^{i\omega_n t} \tag{15}$$

Once we know the N values of the transform $\hat{f}_n$, we can use this expression to evaluate $f(t)$ for any time t. The frequencies ωn are determined by the number of samples taken and by the total sampling time $T = Nh$ as

$$\omega_n = n\frac{2\pi}{Nh} \tag{16}$$

Clearly, the larger we make the time $T = Nh$ over which we sample the function, the smaller will be the frequency steps or resolution. Accordingly,if you want a smooth frequency spectrum, you need to have a small frequency step $2\pi/T$.

Discrete Fourier Transforms (IV)

Lastly, we can simplify this expression to yield a clear computational approach:

$$f_k = \frac{\sqrt{2\pi}}{N} \sum_{n=1}^{N} Z^{-nk} \hat{f}_n \qquad (Z = e^{-2\pi i/N}) \tag{17}$$

$$\hat{f}_n = \frac{1}{\sqrt{2\pi}} \sum_{k=1}^{N} Z^{nk} f_k \qquad (n = 0, 1, \cdots, N) \tag{18}$$

With this formulation, the computer needs to compute only powers of Z.