ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

A mathematical tool: The Discrete Fourier transform

Computational imaging 2024-25

Elena Loli Piccolomini

Dipartimento di Informatica - Scienza e Ingegneria (DISI)

A fundamental mathematical tool: the discrete Fourier transform

Preliminary: complex numbers

Links…

- https://www.mathsisfun.com/numbers/complex-numbers.html
- https://www.cuemath.com/numbers/complex-numbers/

…. Many more!

A fundamental mathematical tool in signal and image processing is the Fourier transform.

The Fourier transform is a continuous transformation bewteen spaces of functions.
The discretized version, the Discrete Fourier Transform, is a function between vector spaces.

The Fourier series

A very clear youtube tutorial:

https://www.youtube.com/watch?v=mgXSevZmjPc

Each periodic function of period 2π

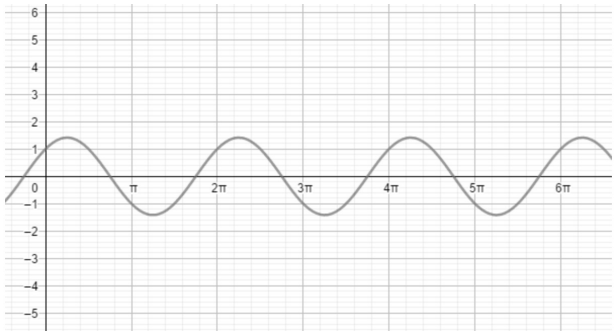
$$f : R \to R$$

can be written as a sum of sin and cos functions with different frequencies :

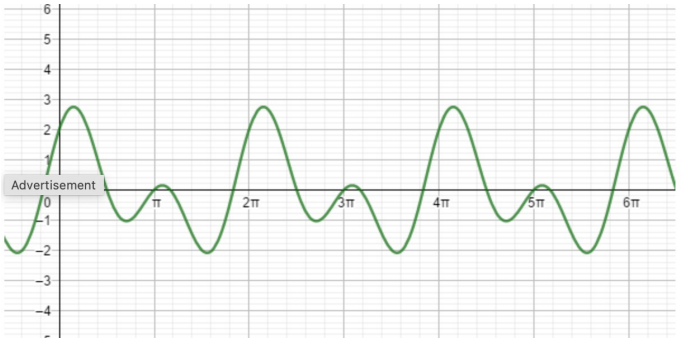
$$f(x) = A_0/2 + \sum_{k=1}^{N} A_k cos(kx) + B_k sin(kx)$$

The Fourier series

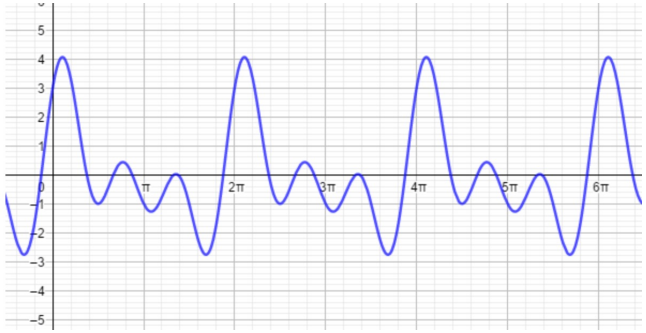
The first components of the Fourier sum (suppose a_0=0, a_k=b_k=1):



K=1

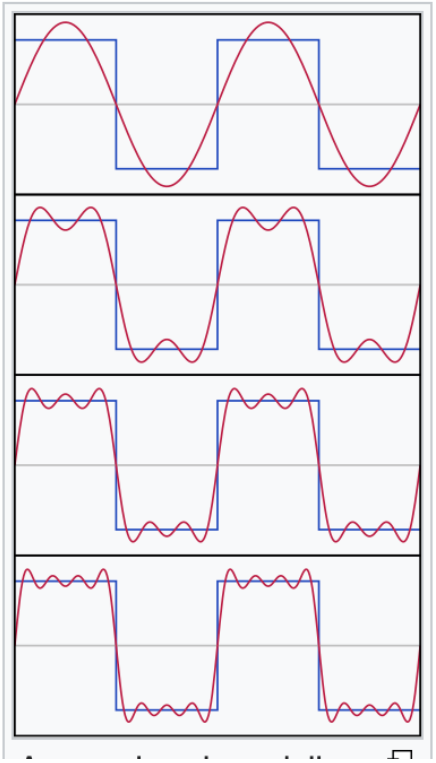


K=2

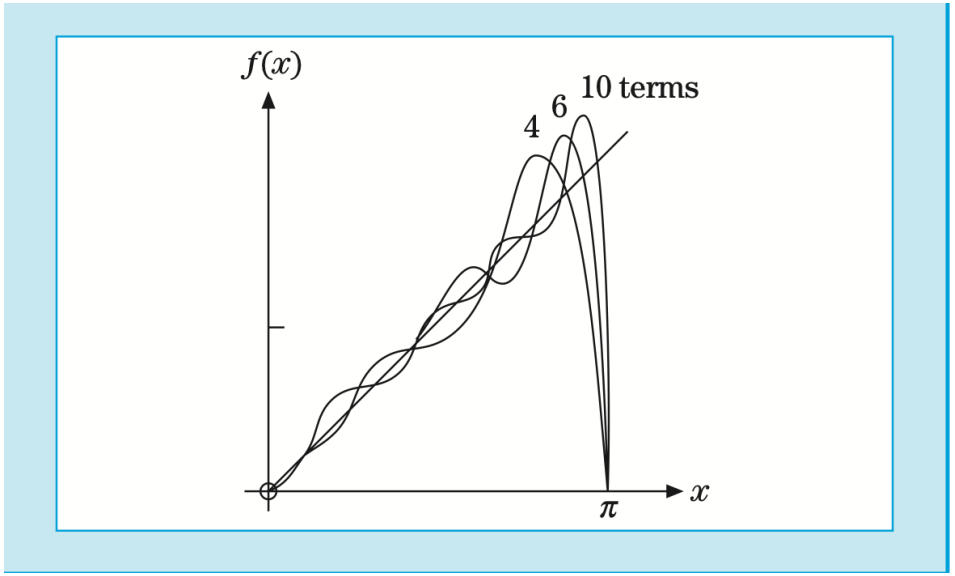


K=3

The Fourier series



Approximation of the square wave with its first five terms of the Fourier sum

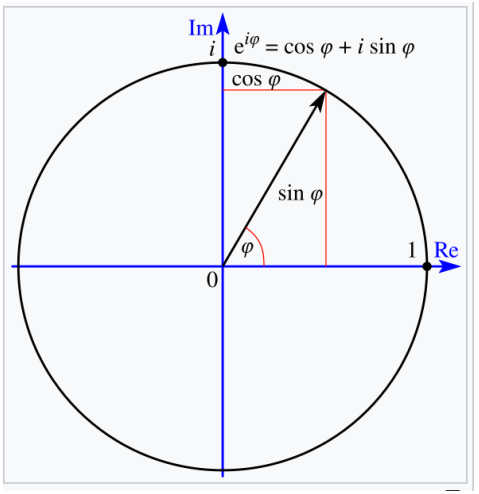


Approximation of the sawtooth wave with its first ten terms of the Fourier sum

The Eulero's Formula :

Eulero's Formula :

$$e^{ix} = cos(x) + isin(x)$$



By using Eulero's formula, we can write the Fourier sum in the equivalent form:

$$f(x) = \sum_{k=-N}^{N} c_i e^{ikx}$$

Where the coefficents c_i are complex numbers and i is the imaginary unit.

The Fourier series for a general function

For a general function with complex values

$$f : R \to C$$

The Fourier sum is generalized as:

$$\sum_{k=-\infty}^{\infty} c_i e^{\frac{i2\pi kx}{T}}$$

The Fourier transform

The Fourier Transfom is a map that transforms a function in another function. Hence it is a map between *function spaces H and K:*

$$\mathcal{F} : H \to K, \quad \mathcal{F}(f) = \hat{f}$$

If H contains periodic functions of period T, that can be represented by means of the Fourier series, the Fourier transform computes the coefficents c_i of the Fourier series.

$$\hat{f}(\omega) = \int_{-T/2}^{T/2} f(x) e^{-\frac{i2\pi x\omega}{T}} dx$$

But the Fourier transform can be applied to any function, not necessarily periodic.

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi x} dx$$

The Inverse Fourier transform

The Inverse Fourier Transfom is the inverse map

$$\mathcal{F}^{-1} : K \to H, \quad \mathcal{F}(\hat{f}) = f$$

Since it is the inverse map, it holds:

$$\mathcal{F}^{-1}(\mathcal{F}(f)) = f$$

The Inverse Fourier Transform reads as (now the indepedent variable ω is a frequency):

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(\omega) e^{i2\pi\omega} d\omega$$

The Fourier transform

- Its most common use is to transform a function that varies over time, such as a sound, into a function related to frequencies, which is often easier to analyze.

- Since applying the transform twice returns to the original function, it can also be used to modify the original function—for example, by "cleaning up" a recording full of noise by removing the higher frequencies, which are mostly associated with it.

- WhWhen we have a discrete set of function values instead of a continuous function, the Fourier transform is replaced by the discrete Fourier transform (DFT).

- It is a transform that converts a finite collection of evenly spaced samples of a function into a collection of coefficients of a linear combination of complex sinusoids, ordered by increasing frequency.

Discrete Fourier Transform 1D

- The Discrete Fourier Transform (DFT) is a mathematical technique used to convert spatial or temporal data into frequency data. Essentially, it takes a finite sequence of equally spaced samples of a function and decomposes it into a sum of sinusoids of different frequencies.

- The DFT provides the frequency domain representation of a signal, which is often more insightful than its time domain representation. By understanding the frequency components of a signal, we can perform various operations such as filtering, signal reconstruction, and more.

- In the context of the DFT, each frequency component obtained from the transformation is expressed as a complex number, comprising both real and imaginary parts. The real component represents the amplitude of the cosine wave in the frequency domain, while the imaginary part corresponds to the amplitude of the sine wave. Together, these components provide a complete representation of the frequency characteristics of a signal.

Discrete Fourier Transform 1D

The Discrete Fourier Transform is a linear invertible function $F : C^N \rightarrow C^N$

Let f be a discrete signal: $f = (f_0, f_1, \ldots f_{N-1})$

The **Discrete Fourier Transform (DFT)** is a complex array of N elements:

$$\hat{f} = (\hat{f}_0, \hat{f}_1, \ldots \hat{f}_N - 1) \qquad \hat{f}_m = \sum_{n=0}^{N-1} f_n e^{-i\frac{2\pi}{N}mn}, \quad m = 0, \ldots N-1$$

It can be written as a matrix-vecor product

Where

$$\hat{f} = Ff$$

$$F_{m,n} = e^{-i\frac{2\pi}{N}mn}$$

N.B. i is the imaginary unit and the DFT is a vector of complex numbers, where the real component represents the amplitude of the cosine wave in the frequency domain, while the imaginary part corresponds to the amplitude of the sine wave.

Inverse Discrete Fourier Transform 1D

The **Inverse Discrete Fourier Transform (IDFT)** is a vector :

$$f_m = \frac{1}{N} \sum_{m=0}^{N-1} \hat{f}_m e^{i \frac{2\pi}{N} mn}, \quad m = 0, \ldots N-1$$

In matrix form:

$$f = F^{-1} \hat{f} \qquad\qquad f = (f_0, f_1, \ldots f_N - 1)$$

N.B. The IDFT should be a vector of real numbers but due to finite arithmetic errors it can have a small imaginary part.

The Fast Fourier Transform (FFT) algorithm is a very efficient algorithms that computes the Direct and Inverse FT transform in O(N log_2(N)) operations, when N=2^p.

Discrete Fourier Transform

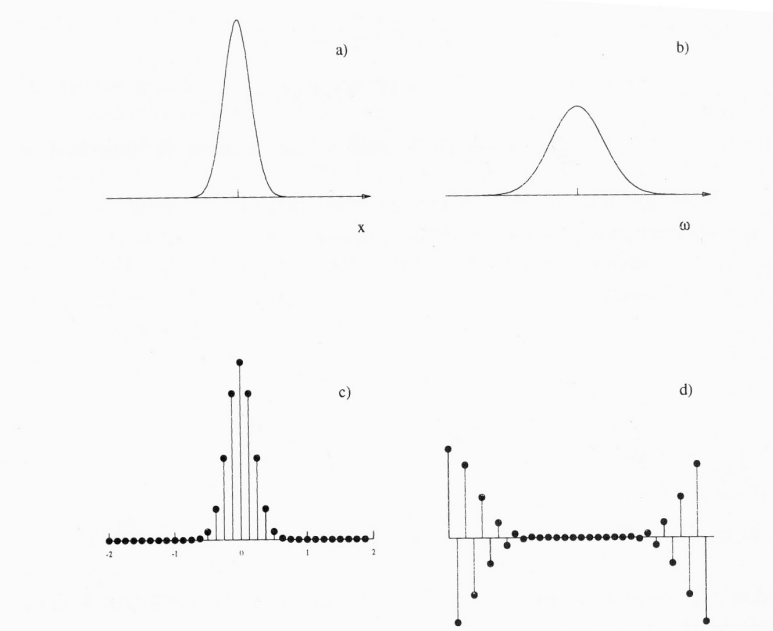


Figure 2.7. Illustrating the relationship between the FT of a function and the DFT o samples: (a) a function $f(x)$ with a Gaussian shape; (b) the FT of the function $f(x)$ the samples of $f(x)$; (d) the DFT of the samples of $f(x)$.

The DFT of a real vector has symmetric values

with respect to the central one.

Discrete Fourier Transform 2D

- DFT:

$$\hat{f}_{k,l} = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f_{m,n} exp\left(-i\frac{2\pi}{N}(km + ln)\right)$$

- IDFT:

$$f_{m,n} = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} \hat{f}_{k,l} exp\left(i\frac{2\pi}{N}(km + ln)\right)$$

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

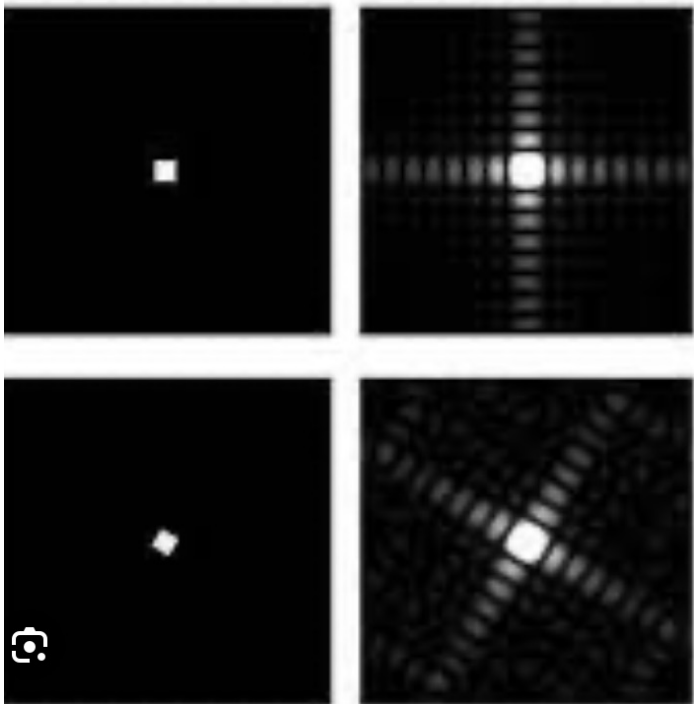
DFT 2D in Python

Python libraries:

- `numpy.fft`
- `scipy.fft`

Functions:

- `Fft (1D)`
- `Ifft (1D)`
- `Fft2 (2D)`
- `Ifft2(2D)`



Images (on the left) and modulus
(or amplitude) of their DFT (on the right)

Properties of DFT

The Fourier transform creates a correspondence between the images and its DFT.

The values in the Fourier domain, usually called frequencies, have some usefule properties
For image manipulation, such as

- Reduce the noise on the image (denoising)
- Reduce the space occupied by the image (compression)

The noise is mainly localized in the high frequencies, i.e. in the edges of the objects in the image.

https://www.youtube.com/watch?v=OOu5KP3Gvx0

Denoising images with DFT

- To denoise an image using a Fast Fourier Transform (FFT) filter, we can use a simple method of applying a low-pass filter in the frequency domain.

- The idea is to suppress high-frequency noise, which typically exists in the higher frequencies of the Fourier spectrum, while preserving the low-frequency components, which generally correspond to the important structure of the image.

- To reduce the ringing artifacts, you can apply a **soft cutoff** or **smooth filter** in the frequency domain. This can be done by using a smoother low-pass filter, such as a **Gaussian filter** or **Butterworth filter**, instead of a sharp circular mask. These filters gradually attenuate the high frequencies, which helps to avoid the abrupt transitions that lead to ringing.

Denoising images with DFT

Algorithm :

1.Load the image:

2.FFT: Apply a 2D FFT to the image (np.fft.fft2), which transforms the image into the frequency domain.

3.Low-pass filter: create a low-pass filter by constructing a circular mask that keeps the low-frequency components (center of the FFT) and removes high-frequency components (outside the mask). The cutoff_frequency controls the size of the central part of the mask that is kept.

4. **Apply the filter**: We apply the mask to the shifted FFT result (fshift * mask), which removes high-frequency noise.

5. **Inverse FFT**: We perform an inverse FFT (np.fft.ifft2) to transform the filtered result back into the spatial domain and obtain the denoised image.

Parameter: The cutoff_frequency defines how aggressive the denoising is. A higher value keeps more details but may allow some noise to persist, while a lower value results in stronger denoising but may also remove fine details.

Compressing images with DFT

The Discrete Fourier Transform (DFT) plays a significant role in image compression, a crucial process for reducing file sizes while maintaining image quality.

By transforming the spatial representation of an image into the frequency domain, the DFT allows us to identify and retain only the most significant frequency components, discarding less important data without a substantial loss of detail.

Techniques such as the JPEG compression algorithm leverage this principle, using transformations similar to the DFT to efficiently encode image data.

By focusing on the most critical frequencies, the DFT enables significant reductions in image size, allowing for faster transmission and storage while preserving essential features.

Compressing images with DFT

Algorithm :

1. Read the image

2. FFT: Apply a 2D Fast Fourier Transform (np.fft.fft2) to convert the image into frequency space.

3.**Compression**: We discard the high-frequency components by setting their magnitude to zero if they are below a threshold. This allows us to compress the image by removing less significant frequency components.

4. Reconstruction: The image is reconstructed by combining the modified magnitude with the original phase, and then performing an inverse FFT (np.fft.ifft2).

5. **Display**: Finally, the original and compressed images are displayed side by side for comparison.

Parameter: threshold. This is the fraction of the maximum magnitude below which the frequency components are discarded. You can adjust this to control the compression rate. A lower value will preserve more high-frequency details and result in less compression, while a higher value will discard more frequencies, resulting in more compression

Fundamentals of cyclic convolution

The fundamental theorem of convolution in 1D.

Theorem

Suppose $f = (f_0, f_1, \ldots f_{N-1})$ a signal extended with periodic boundary conditions,
K a convolution kernel of size extended with periodical conditions, and g the result of the convolution of K and f:

$$g = k * f$$

Then:

$$\hat{g} = \hat{K}\hat{f}$$

The theorem trivially extends for images in 2D.