Contents

- Fourier Transform
- Convolution and Correlation
  - Problems

Fourier Transform

How to Represent Signals?

- Option 1: Taylor series represents any function using polynomials.

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}$$
$$(x-a)^2 + \frac{f^{(3)}(a)}{3!}(x-a)^3 + \ldots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \ldots.$$

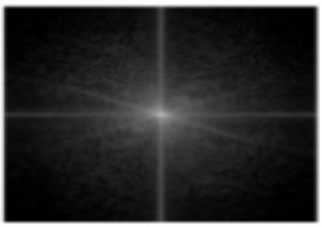
- Polynomials are not the best - unstable and not very physically meaningful.

- Easier to talk about “signals” in terms of its “frequencies” (how fast/often signals change, etc).

Jean Baptiste Joseph Fourier (1768-1830)

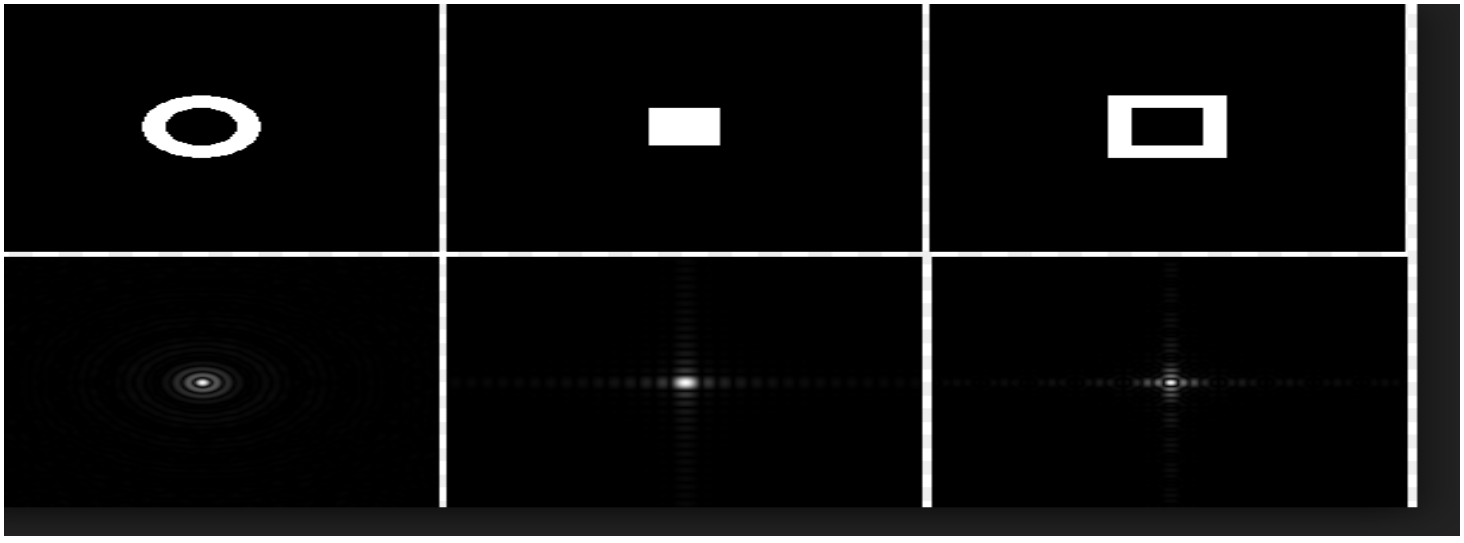
- Had crazy idea (1807):
- **Any** periodic function can be rewritten as a weighted sum of Sines and Cosines of different frequencies.
- Don't believe it?
  - Neither did Lagrange, Laplace, Poisson and other big wigs
  - Not translated into English until 1878!
- But it's true!
  - called Fourier Series
  - Possibly the greatest tool used in Engineering

Fourier Transform/2



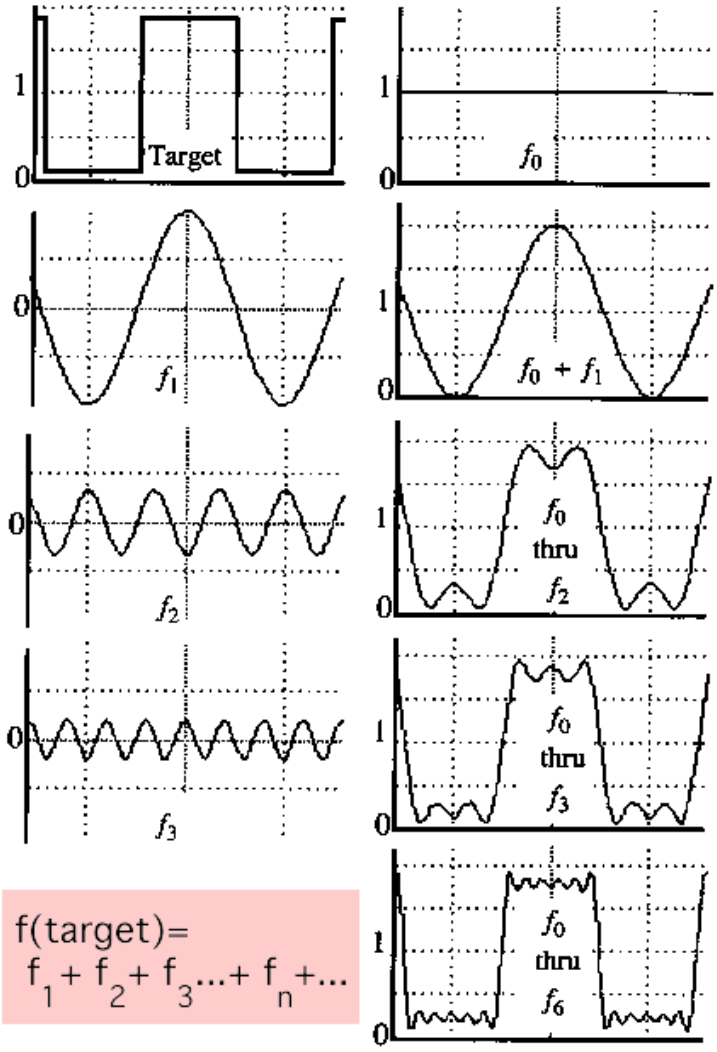
Original image

Gaussian blur with sigma value of 3



A Sum of Sinusoids

- Our building block:
- Add enough of them to get any signal *f(x)* you want!
- How many degrees of freedom?
- What does each control?
- Which one encodes the coarse vs. fine structure of the signal?



f(target)=
$f_1 + f_2 + f_3 ... + f_n + ...$

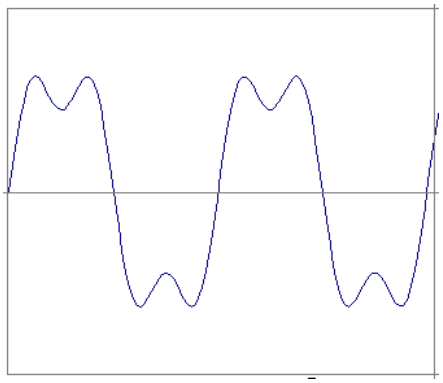
Fourier Transform

- We want to understand the frequency ω of our signal. So, let's reparametrize the signal by ω instead of *x*:

- For every ω from 0 to inf, ***F(ω)*** holds the amplitude *A* and phase φ of the corresponding sine
  - How can *F* hold both? Complex number trick!

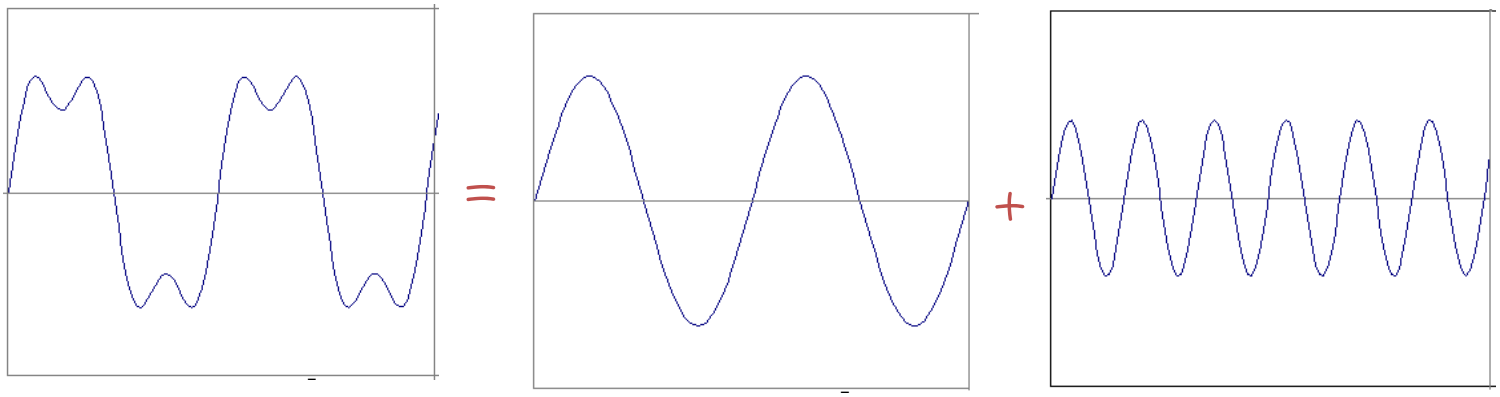
Time and Frequency

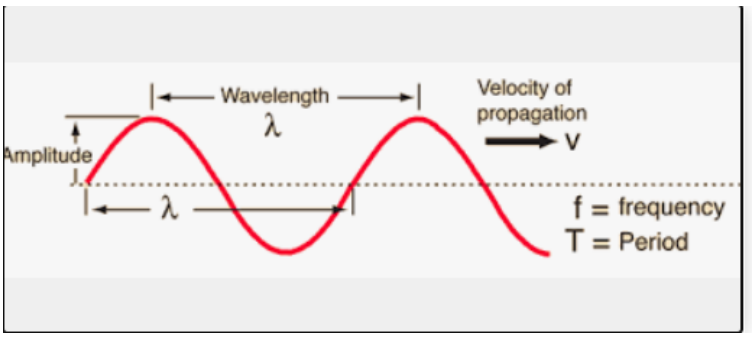
- example : $g(t) = \sin(2\pi f t) + (1/3)\sin(2\pi (3f) t)$



Time and Frequency

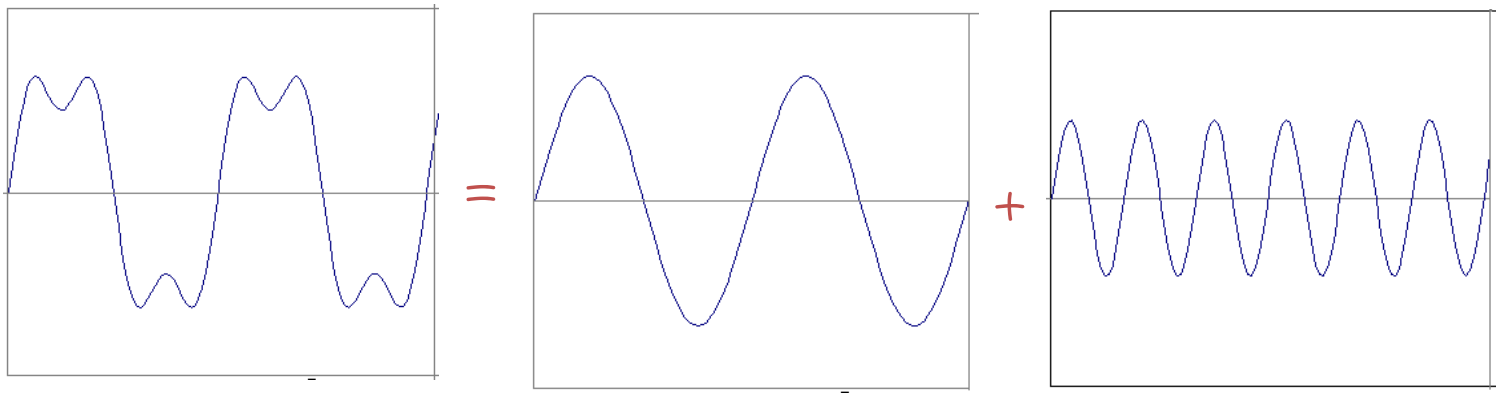
- example : $g(t) = \sin(2\pi f t) + (1/3)\sin(2\pi (3f) t)$

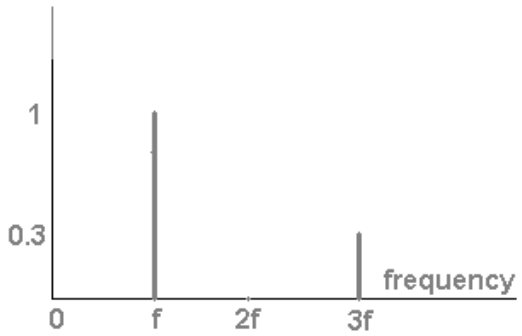




Frequency Spectra

- example : $g(t) = \sin(2\pi f t) + (1/3)\sin(2\pi (3f) t)$





EULER's FORMULA

$$e^{i\theta} = \cos\theta + i \sin\theta$$

$$e^{i\omega t} = \cos\omega t + i \sin\omega t$$

$$\text{where } i = \sqrt{-1}$$

- Euler's formula relates the complex exponential to the complex number whose real part is the cosine function and whose imaginary part is the sine function.
- So the complex exponential is a convenient way to deal with both the sine and the cosine of an argument simultaneously

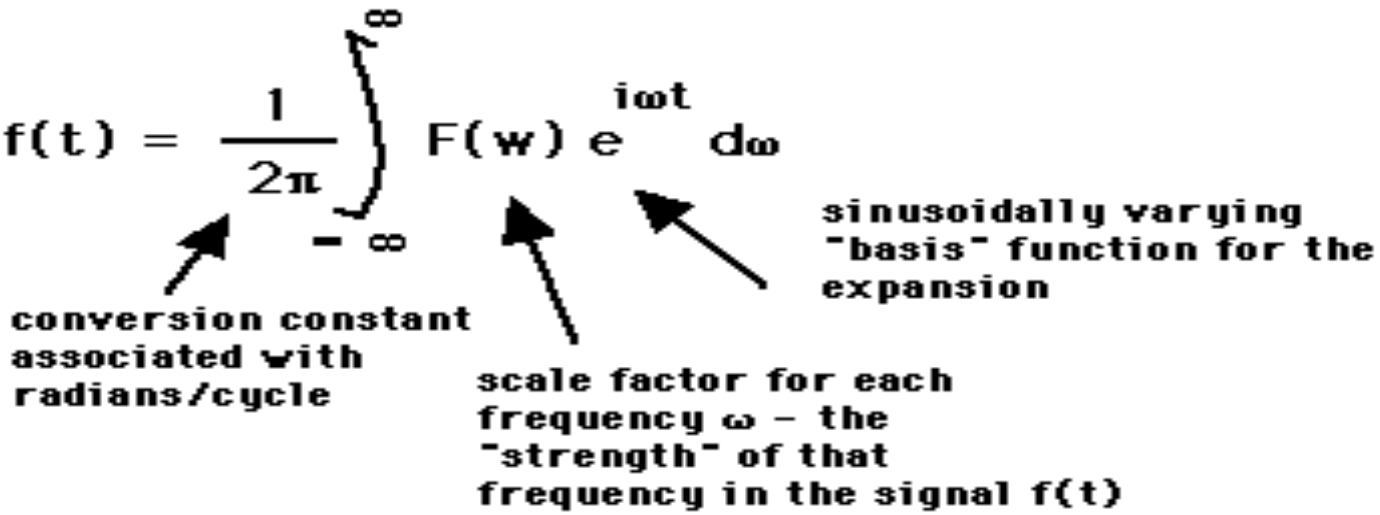
Fourier Transform - Basics

- The complex exponential allows us to deal with the sinusoidal function and phase angles in a functional form which is often easy to manipulate - the exponential form.

- We can represent a time-dependent signal as an expansion of its frequency components.

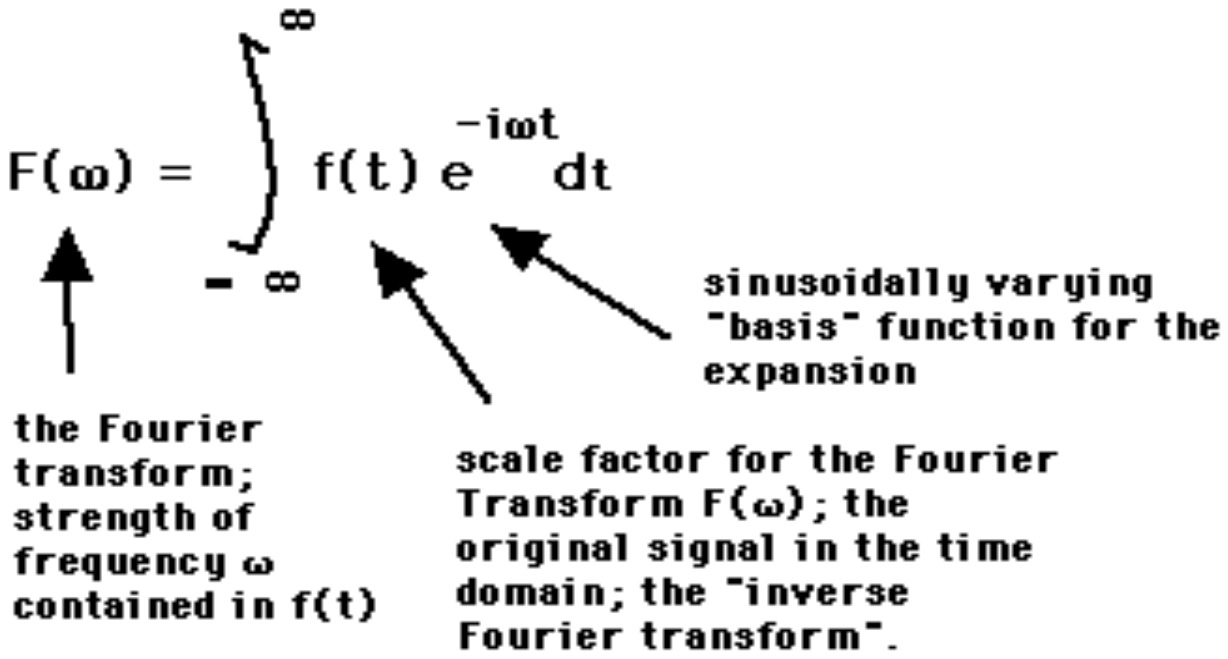
- Time signal contains a certain amount of low frequencies, another amount of medium frequencies, and yet a different amount of high frequencies.

Time signal can be expanded into an infinitely divisible spectrum of frequencies in the following



- Scale factor F(w) for each of the frequencies w is known as the "Fourier Transform" of the original signal –
- It gives the amount of each frequency found in the signal f(t).

Fourier transform F(w) is determined from the original signal f(t) in the following



It is because of this relationship between F(w) and f(t) that we refer to the original time signal as the "Inverse" Fourier transform of F(w).

Fourier Transform – more formally

Represent the signal as an infinite weighted sum of an infinite number of sinusoids

Note:

Arbitrary function ⟶ Single Analytic Expression

Spatial Domain (*x*) ⟶ Frequency Domain (*u*)

(Frequency Spectrum *F(u)*)

Inverse Fourier Transform (IFT)

Fourier transform
and
inverse Fourier transform

direct

inverse

Fourier transform
and
inverse Fourier transform

direct

from the number domain
to the frequency domain

inverse

Fourier transform
and
inverse Fourier transform

direct

from the frequency domain
to the number domain

inverse

Fourier Transform

- Also, defined as:

Note:

- Inverse Fourier Transform (IFT)

Fourier Transform and Convolution

Let

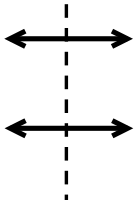
Then

Convolution in spatial domain

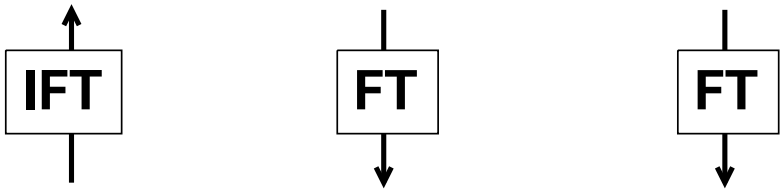
Multiplication in frequency domain

Fourier Transform and Convolution

Spatial Domain *(x)* | Frequency Domain *(u)*



So, we can find *g(x)* by Fourier transform



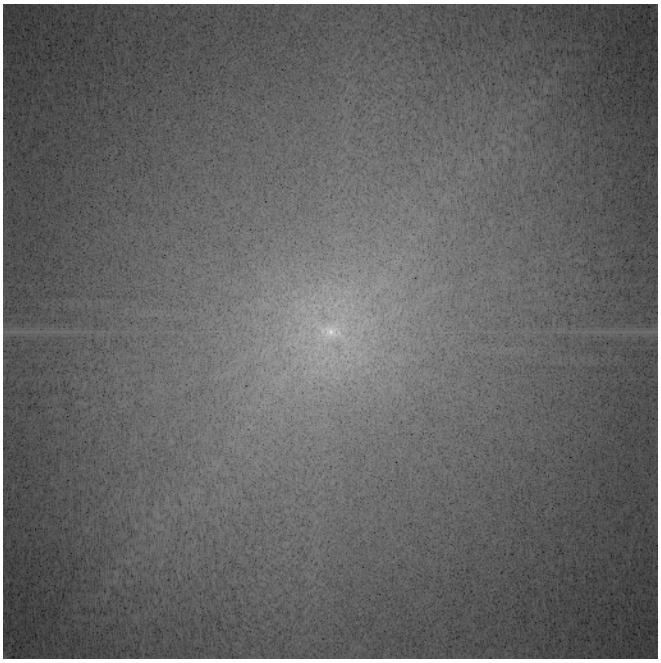
Properties of Fourier Transform

	Spatial Domain *(x)*	Frequency Domain *(u)*
Linearity		
Scaling		
Shifting		
Symmetry		
Conjugation		
Convolution		
Differentiation		

Note that these are derived using frequency ( )

Image Processing in the Fourier Domain

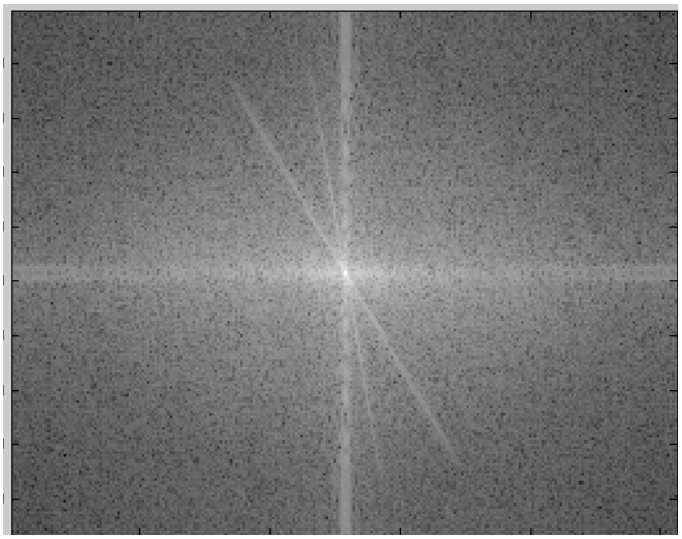
Magnitude of the FT



Does not look anything like what we have seen

Image Processing in the Fourier Domain

Magnitude of the FT



Does not look anything like what we have seen

Convolution

$$g(m,n) = h(m,n) * f(m,n)$$

- Linear filtering of an image is accomplished through an operation called *convolution.*
- *Convolution is a* neighborhood operation in which each output pixel is the weighted sum of neighboring input pixels.
- The matrix of weights is called the *convolution kernel, also known as the filter.*
- *A convolution kernel is a* correlation kernel that has been rotated 180 degrees.

Convolution contd

- Convolution is both (a) commutative and (b) associative

The 1D convolution is defined by

$$(f * g)(x) = \int_{-\infty}^{\infty} f(x)g(x-t)\,dt$$

If the masks are not flipped, then this operation would simply be Correlation

$$(f \star g)(x) = \int_{-\infty}^{\infty} f^*(x)g(x+t)\,dt$$

For example, suppose the image is

```
A = [17  24   1   8  15
     23   5   7  14  16
      4   6  13  20  22
     10  12  19  21   3
     11  18  25   2   9]
```

and the convolution kernel is

```
h = [8   1   6
     3   5   7
     4   9   2]
```

The following figure shows how to compute the (2,4) output pixel using these steps:

How to perform Convolution ?

- Algorithm

1. Rotate the convolution kernel 180 degrees about its center element.
2. Slide the center element of the convolution kernel so that it lies on top of the (2,4) element of A.
3. Multiply each weight in the rotated convolution kernel by the pixel of A underneath.
4. Sum the individual products from step 3.

Convolution

Convolution kernel, ω

1	-1	-1
1	2	-1
1	1	1

Rotate 180º

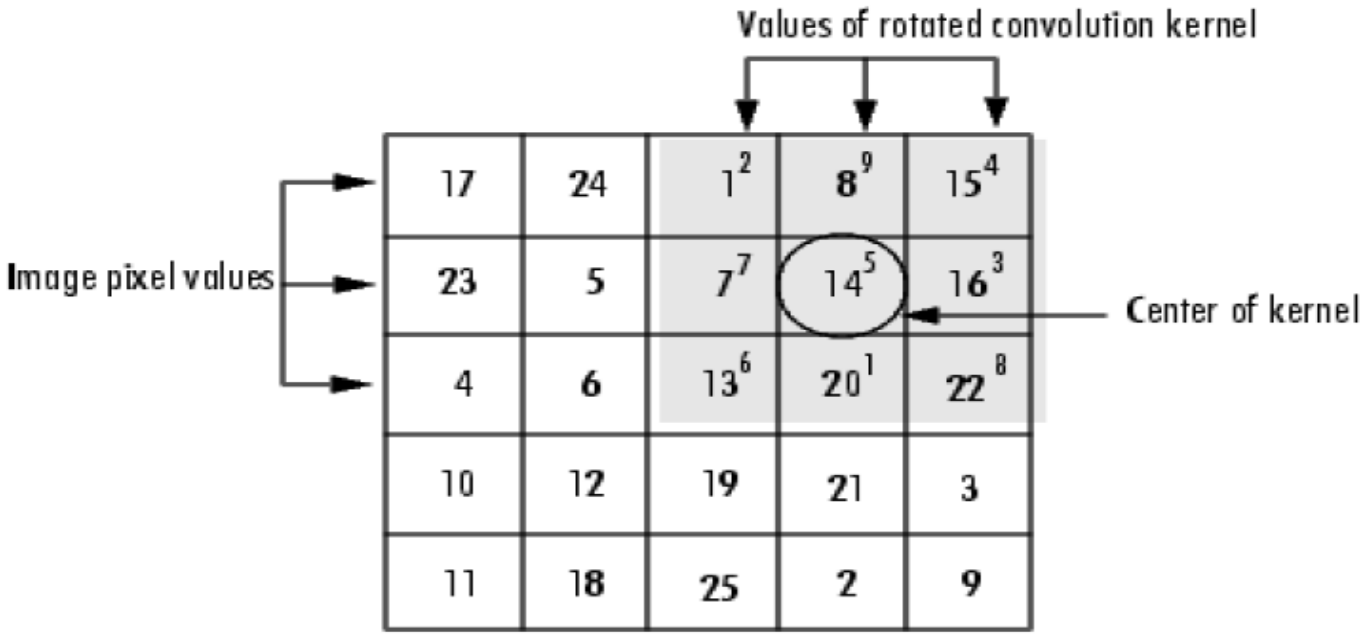
1	1	1
-1	2	1
-1	-1	1

Input Image, f

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

Hence the (2,4) output pixel is

$$1 \cdot 2 + 8 \cdot 9 + 15 \cdot 4 + 7 \cdot 7 + 14 \cdot 5 + 16 \cdot 3 + 13 \cdot 6 + 20 \cdot 1 + 22 \cdot 8 = 575$$



Computing the (2,4) Output of Convolution

Correlation

- Correlation is a measure of similarity.
- The operation called *correlation is closely related to convolution.*
- *In correlation, the value of an output pixel is also* computed as a weighted sum of neighboring pixels.
- The difference is that the matrix of weights, in this case called the *correlation kernel, is not rotated during the computation.*

Correlation - Algorithm

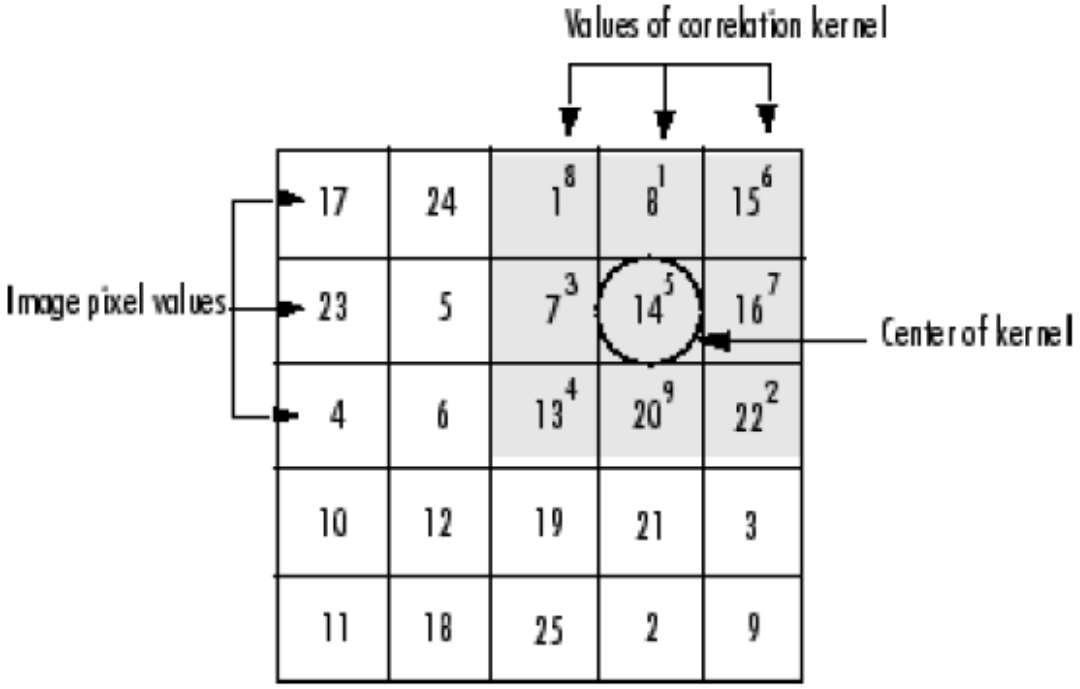
- The following figure shows how to compute the (2,4) output pixel of the correlation of A, assuming h is a correlation kernel instead of a convolution kernel, using these steps:

1. Slide the center element of the correlation kernel so that lies on top of the (2,4) element of A.
2. Multiply each weight in the correlation kernel by the pixel of A underneath.
3. Sum the individual products.

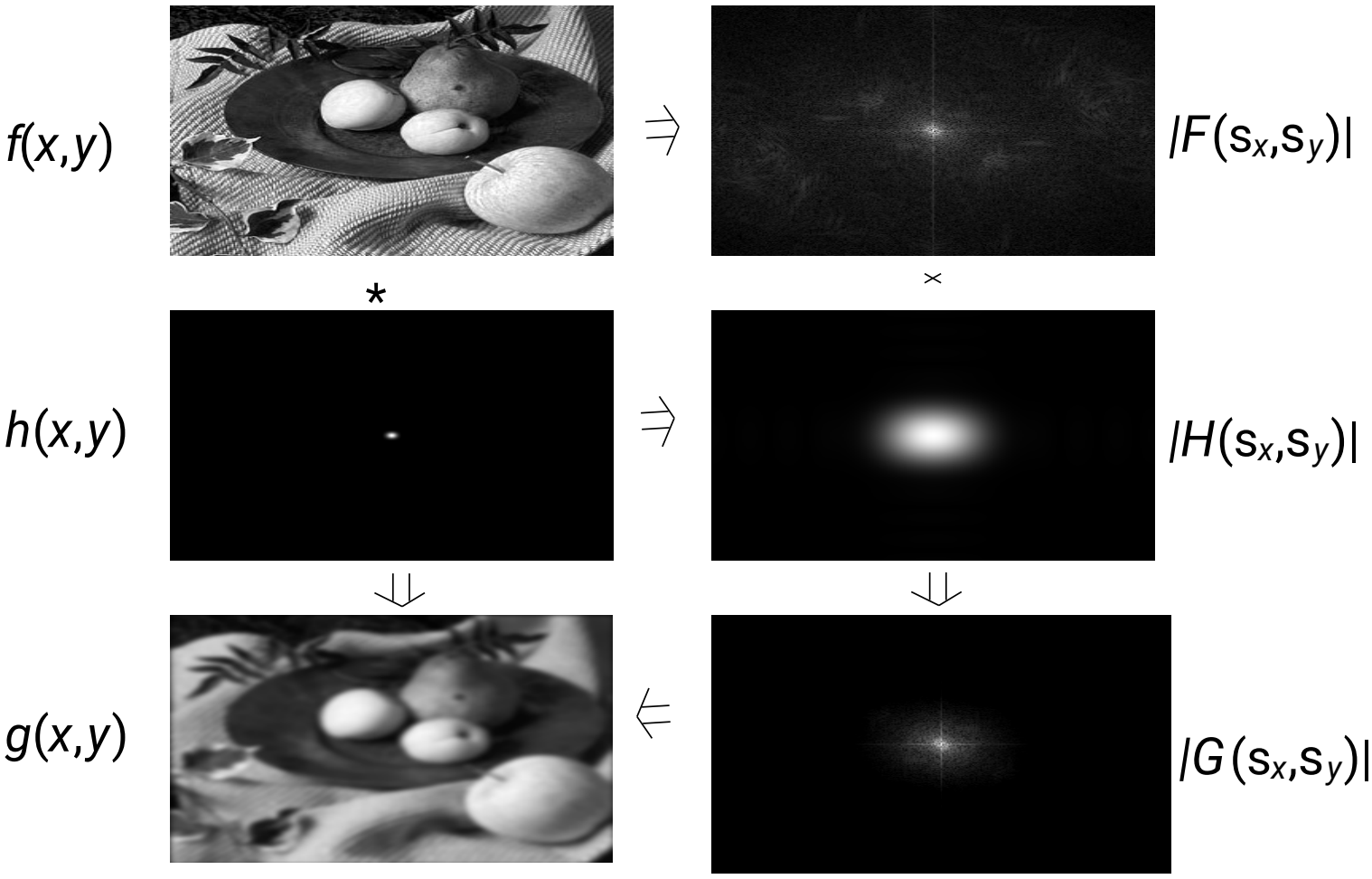
The (2,4) output pixel from the correlation is

$$1 \cdot 8 + 8 \cdot 1 + 15 \cdot 6 + 7 \cdot 3 + 14 \cdot 5 + 16 \cdot 7 + 13 \cdot 4 + 20 \cdot 9 + 22 \cdot 2 = 585$$

Computing the (2,4) Output of Correlation



Convolution is Multiplication in Fourier Domain



Most information at low frequencies!