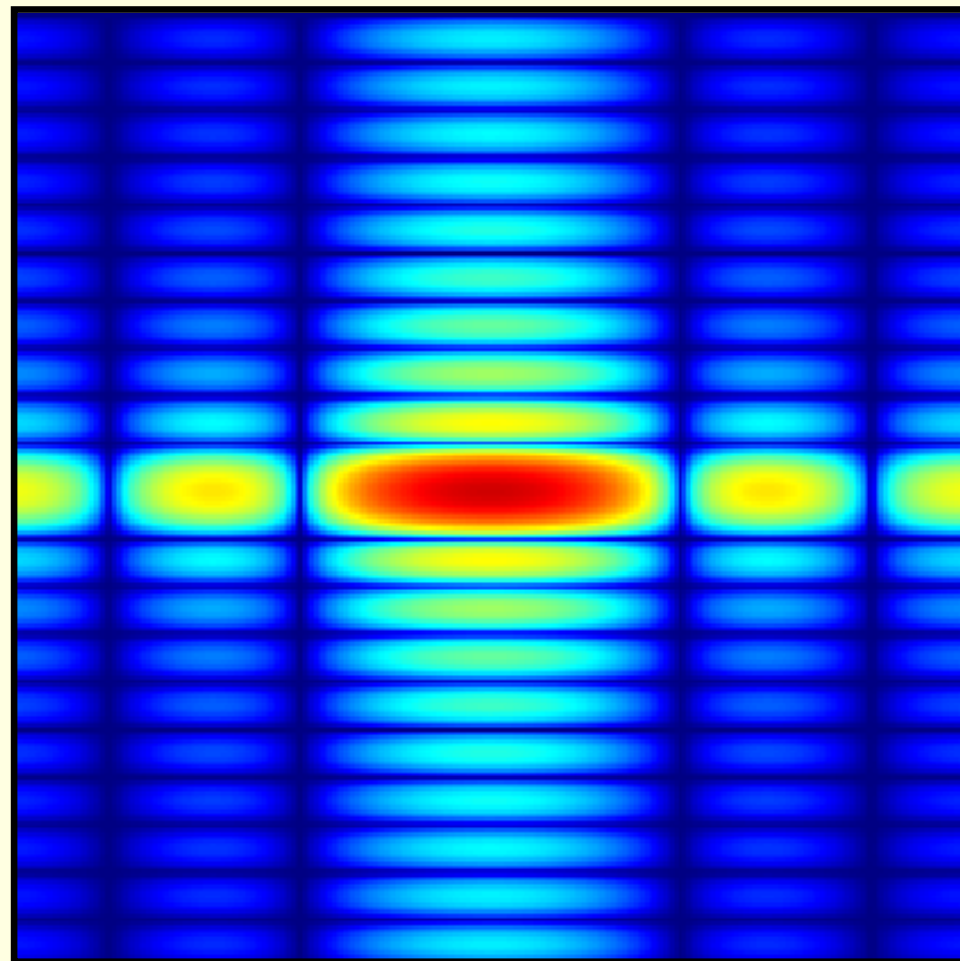


# Fourier transform of images

---



FFT



# Fourier transform

Joseph Fourier has put forward an idea of representing signals by a series of harmonic functions

$$f(x) = \int_{-\infty}^{\infty} \underbrace{F(u)}_{\text{Fourier coefficients}} e^{j2\pi ux} du$$

**inverse**

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} dx$$

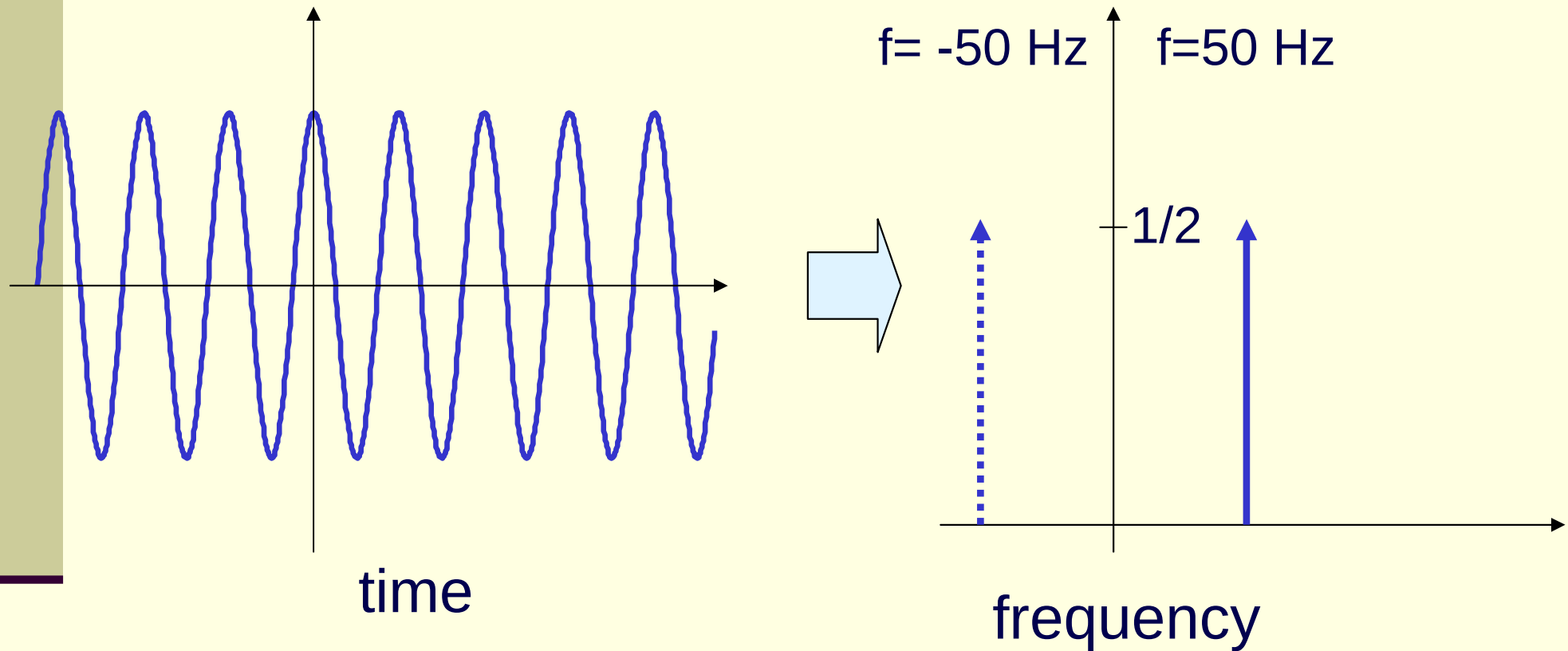
**forward**

Fourier coefficients



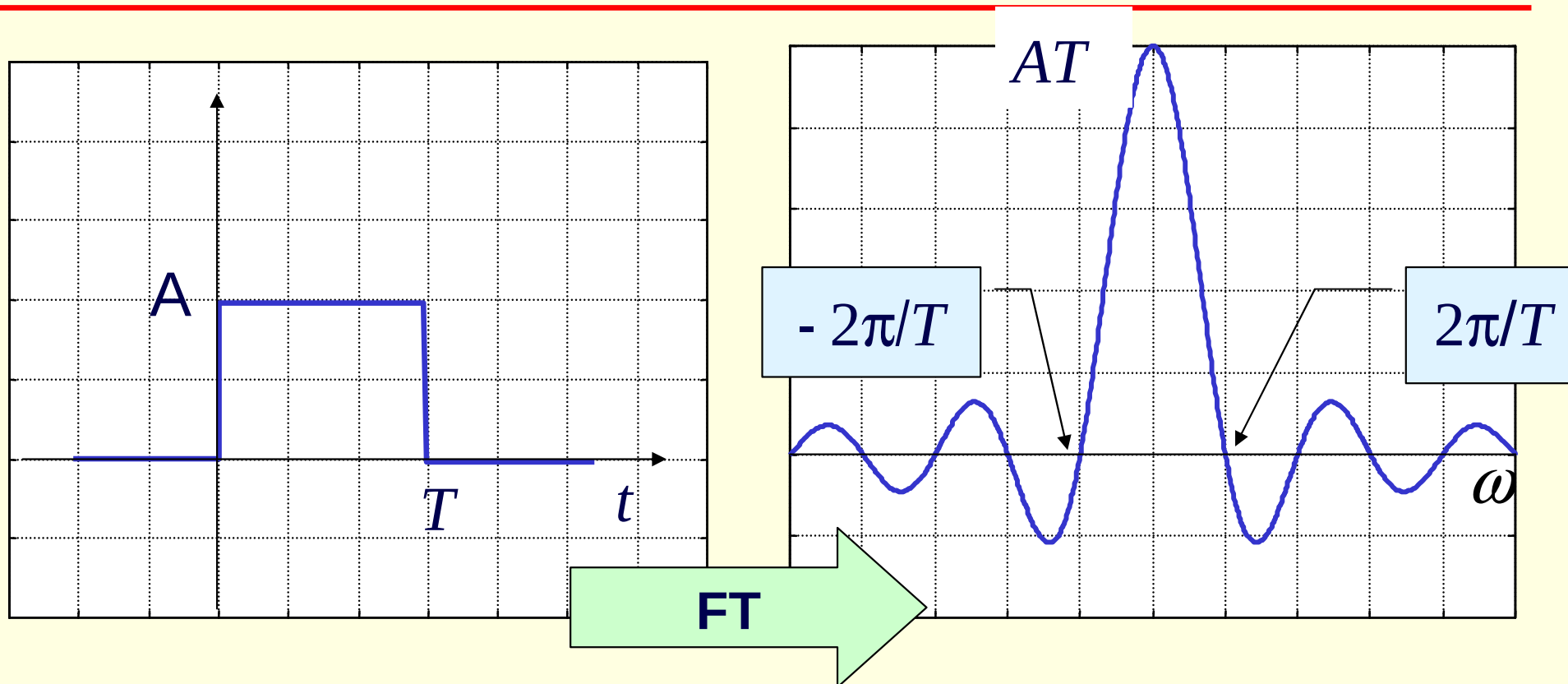
Joseph Fourier  
(1768-1830)

# Fourier transform - example



$$X(j\omega) = \int_{-\infty}^{+\infty} \cos(\omega_0 t) e^{-j\omega t} dt = \frac{1}{2} (\delta(\omega - \omega_0) + \delta(\omega + \omega_0))$$

# Fourier transform - example

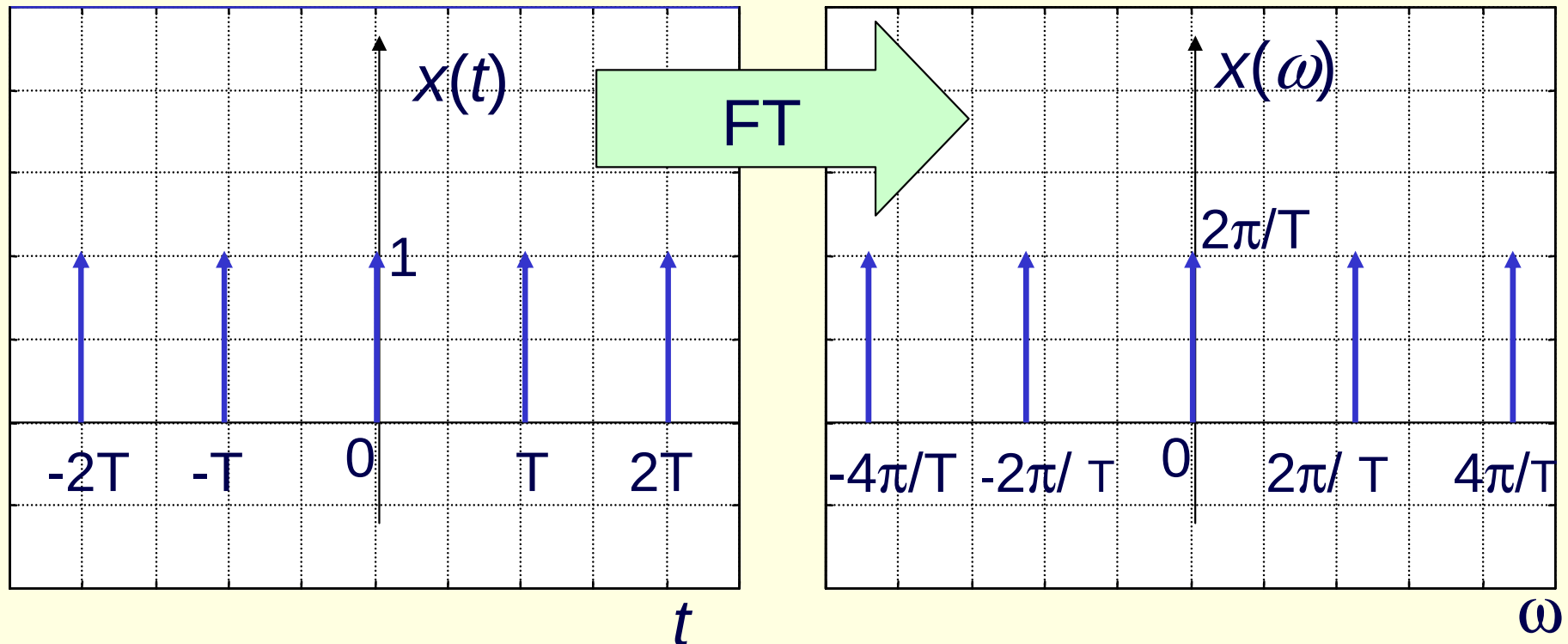


$$f(t) = \begin{cases} 1, & t < T \\ 0, & t > T \end{cases}$$

$$F(\omega) = A \int_0^T e^{-j\omega t} dt = \frac{2A}{\omega} \sin\left(\frac{\omega T}{2}\right) e^{-j\frac{\omega T}{2}}$$

$$|F(\omega)| = ?$$

# Fourier transform - example



A series of Dirac pulses

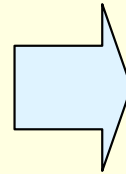
$$\delta_T(t) = \sum_{k=-\infty}^{+\infty} \delta(t - kT) \quad \leftrightarrow \quad \omega_s \sum_{k=-\infty}^{+\infty} \delta(\omega - k\omega_0)$$

$$\omega_s = \frac{2\pi}{T}$$

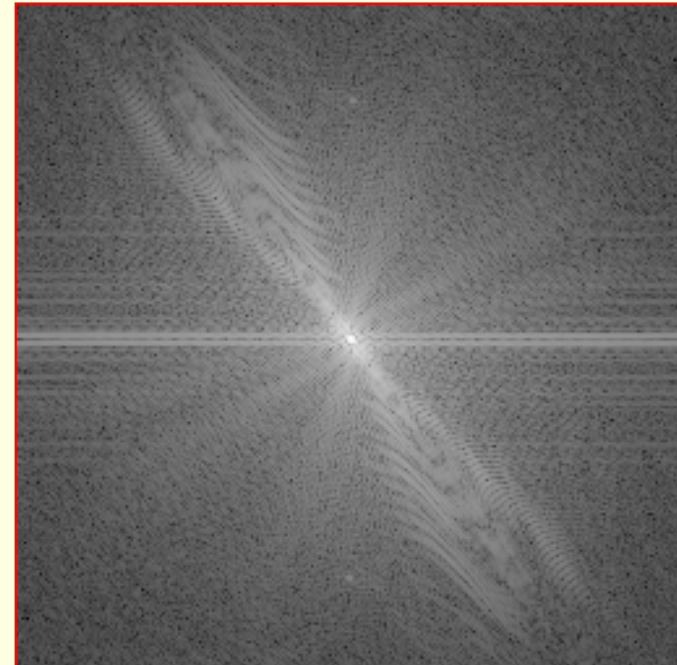
# Fourier transform of images

---

Monochrome image



Fourier spectrum



Why do we convert images (signals) to spectrum domain?

# Fourier transform of images

---

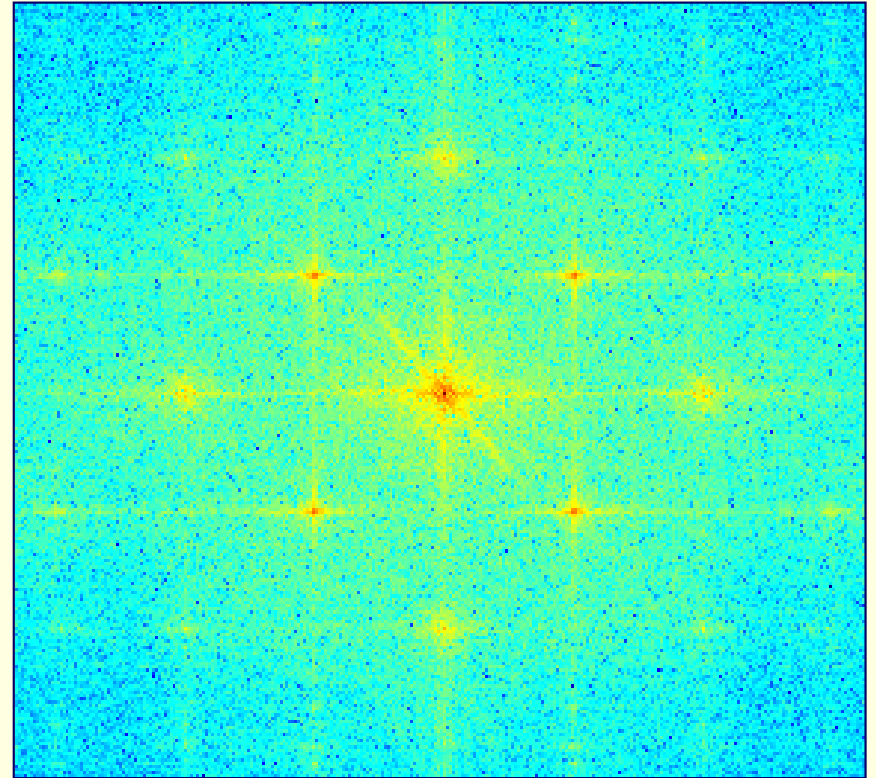
## Why do we convert images to spectrum domain?

1. For **exposing image features** not visible in spatial domain, eg. periodic interferences
2. For achieving more compact image representation (coding), eg. **JPEG, JPEG2000**
3. For **designing digital filters**
4. For fast processing of images, eg. **digital filtering of images** in spectrum domain

# Fourier transform of images

---

## 1. Detection of image features, eg. periodic interferences



# Fourier transform of images

---

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-j2\pi(ux+vy)} dx dy \quad \text{forward}$$

$$f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u,v) e^{j2\pi(ux+vy)} du dv \quad \text{inverse}$$

Euler equations?

$$\cos \omega_0 t = \frac{1}{2} (e^{j\omega_0 t} + e^{-j\omega_0 t})$$

$$\sin \omega_0 t = \frac{1}{2j} (e^{j\omega_0 t} - e^{-j\omega_0 t})$$

# Amplitude and phase spectrum of the Fourier transform of images

---

$$F(u, v) = |F(u, v)| e^{-j \arg[F(u, v)]}$$

$$|F(u, v)| = \sqrt{\operatorname{Re}(F(u, v))^2 + \operatorname{Im}(F(u, v))^2}$$

$$\arg(F(u, v)) = \arctan \frac{\operatorname{Im}(F(u, v))}{\operatorname{Re}(F(u, v))}$$

# The Discrete FT of images

---

$$F(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux+vy)/N}$$

$$\text{dla } u,v = 0,1,\dots,N-1$$

$$f(x,y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u,v) e^{+j2\pi(ux+vy)/N}$$

$$\text{dla } x,y = 0,1,\dots,N-1$$

Number of computations  
for 512x512 image?

# 1D computational example

$$f(x) = [1 \ 3 \ 4 \ 4]$$

$$N = 4$$

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

$$F(u) = \frac{1}{N} \sum_{x=0}^{x=3} f(x) e^{-j2\pi ux/N}$$

$$F(0) = \frac{1}{N} \sum_{x=0}^{N-1=3} f(x) e^{-j2\pi 0x/N} = \frac{1}{4} [f(0) + f(1) + f(2) + f(3)] =$$

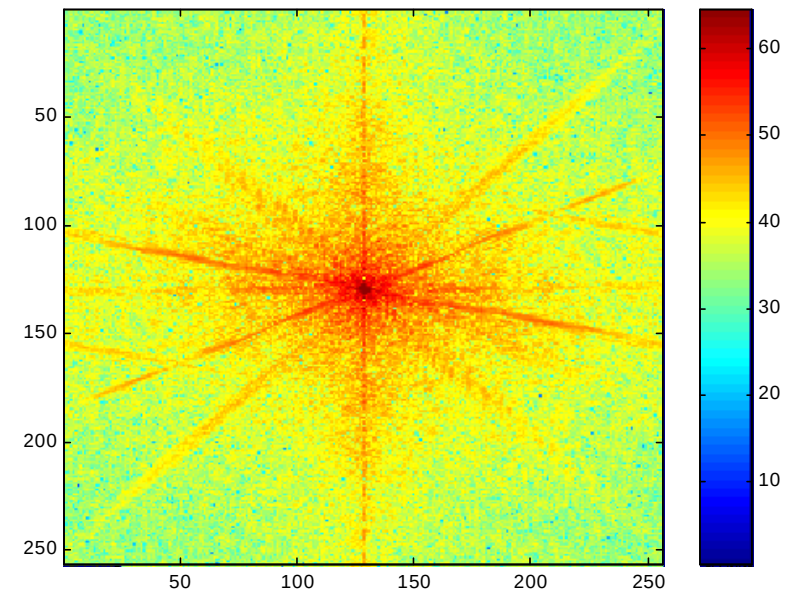
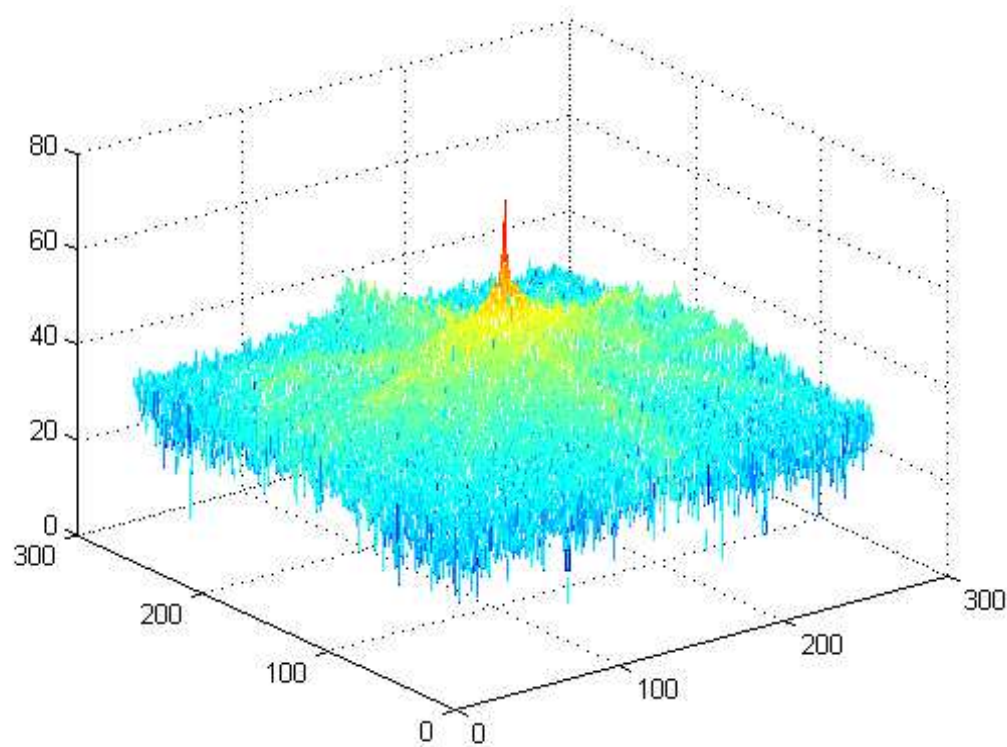
$$= \frac{1}{4} [1 + 3 + 4 + 4] = 3$$

$$F(1) = \dots\dots\dots = \frac{1}{4} (-3 + j)$$

$$F(2) = \dots\dots\dots = -\frac{1}{4} (2)$$

$$F(3) = \dots\dots\dots = -\frac{1}{4} (3 + j)$$

# Fourier amplitude spectrum

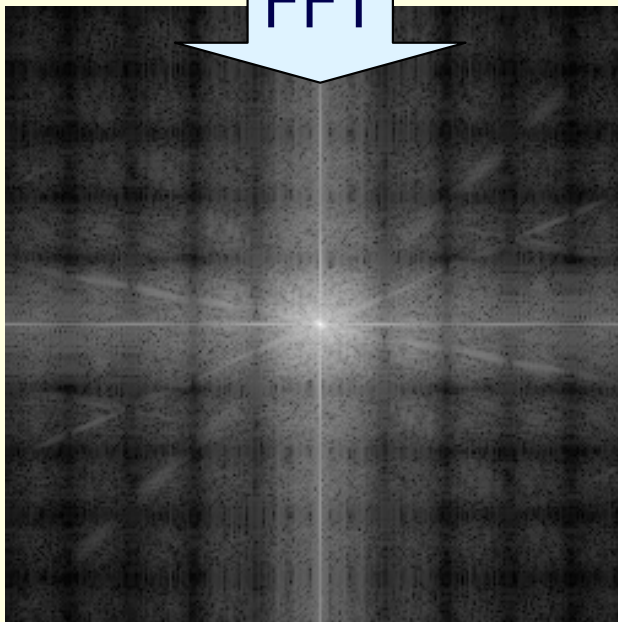


# Fourier amplitude spectrum

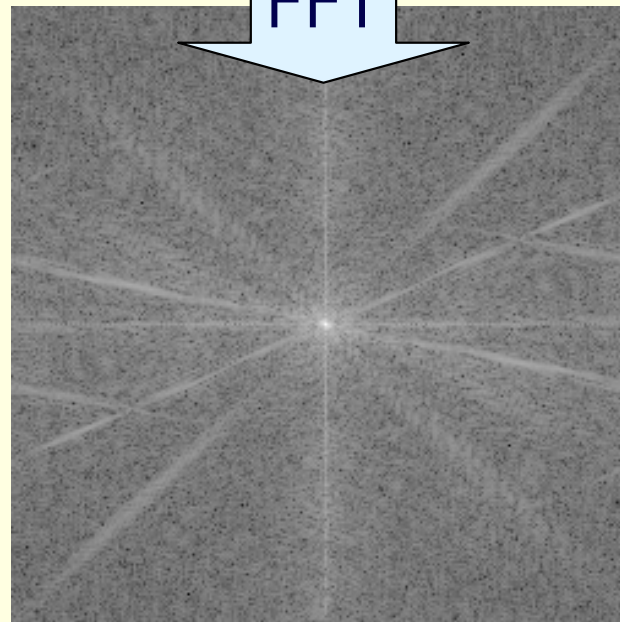
---



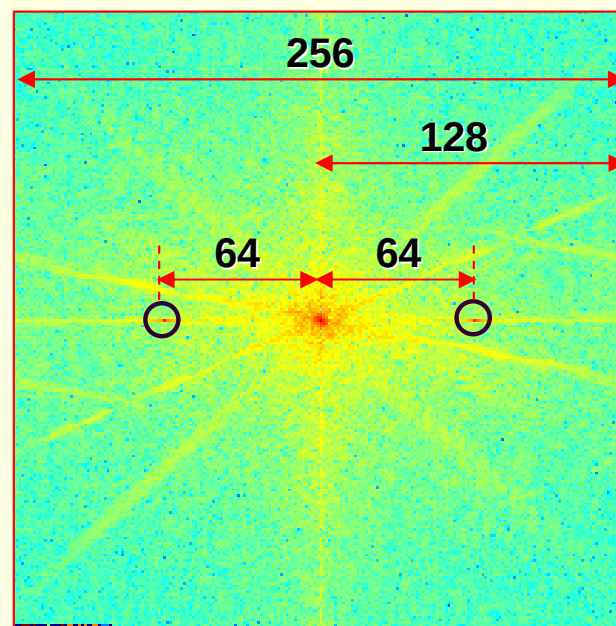
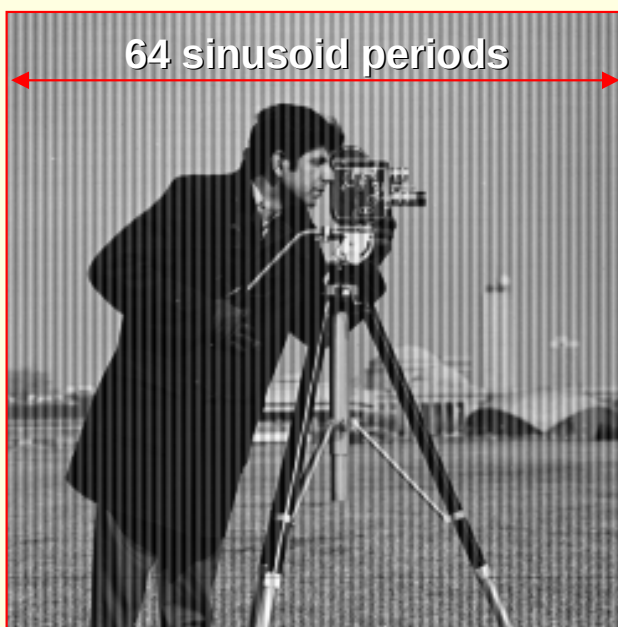
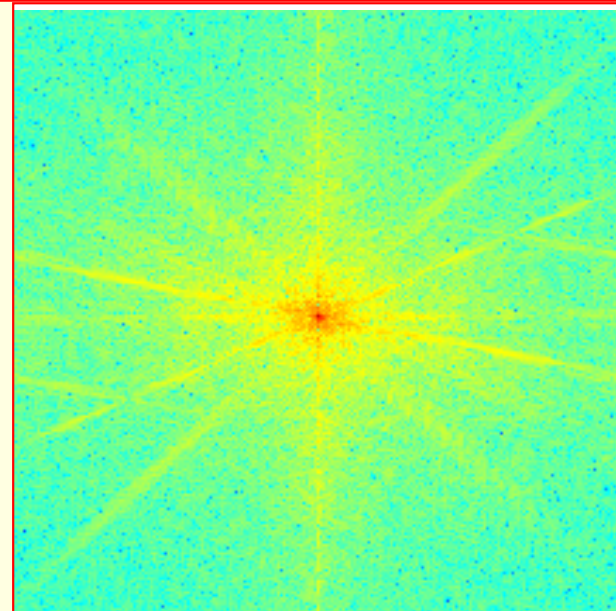
FFT



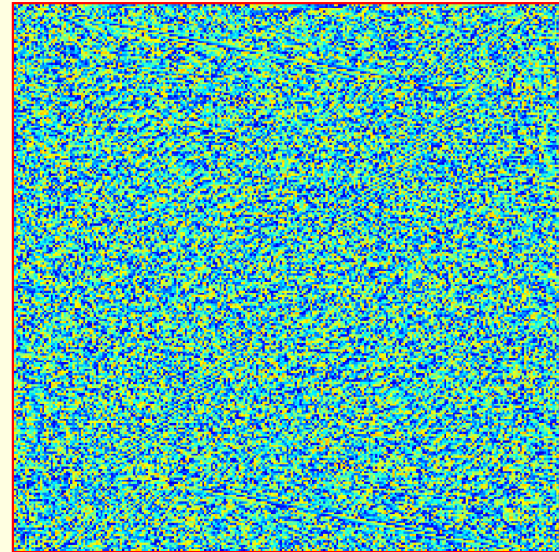
FFT



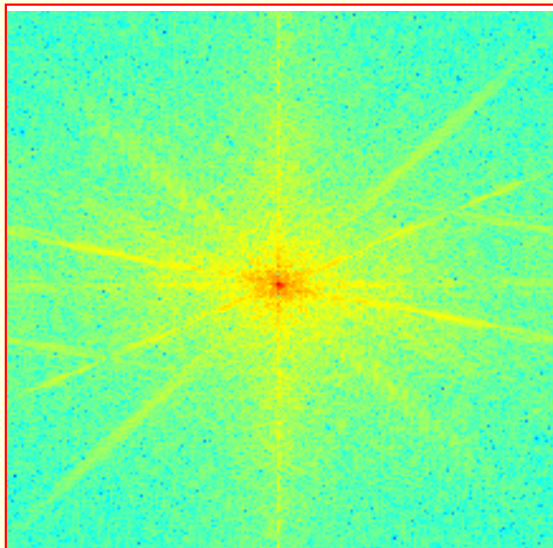
# Detection of periodic distortions



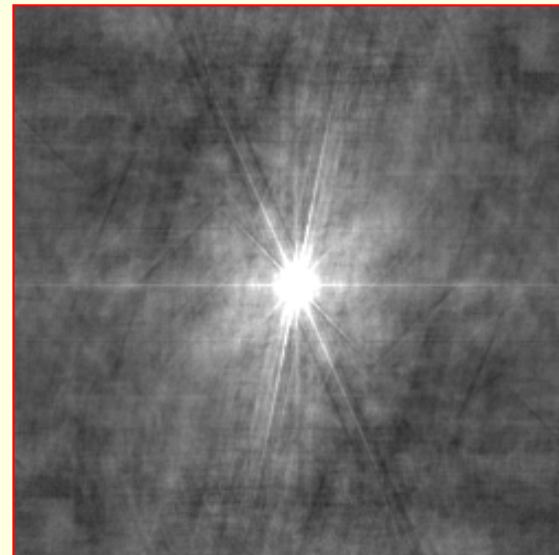
# Fourier phase spectrum of an image



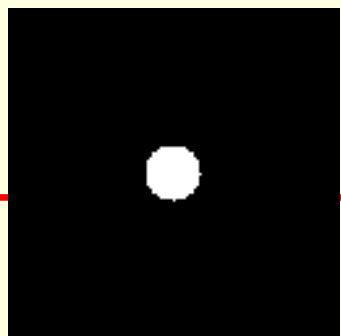
$$\arg[F(u,v)]$$



$$|F(u,v)|$$

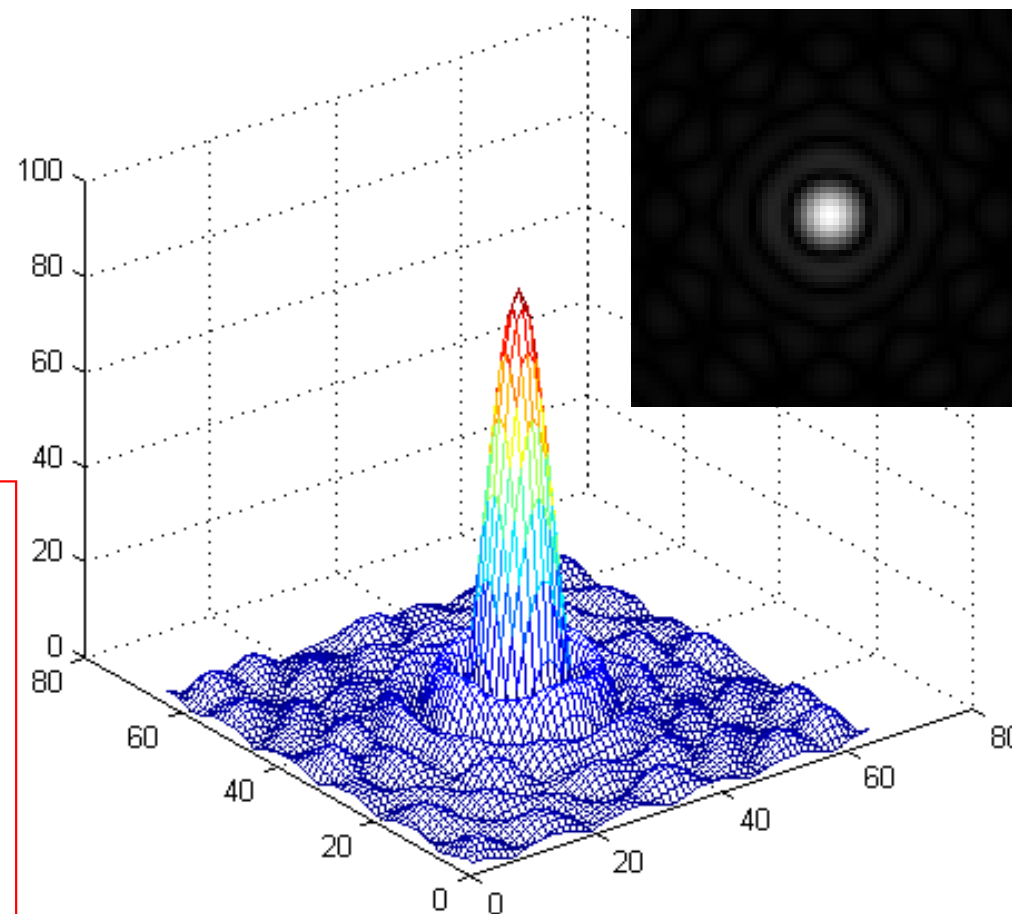
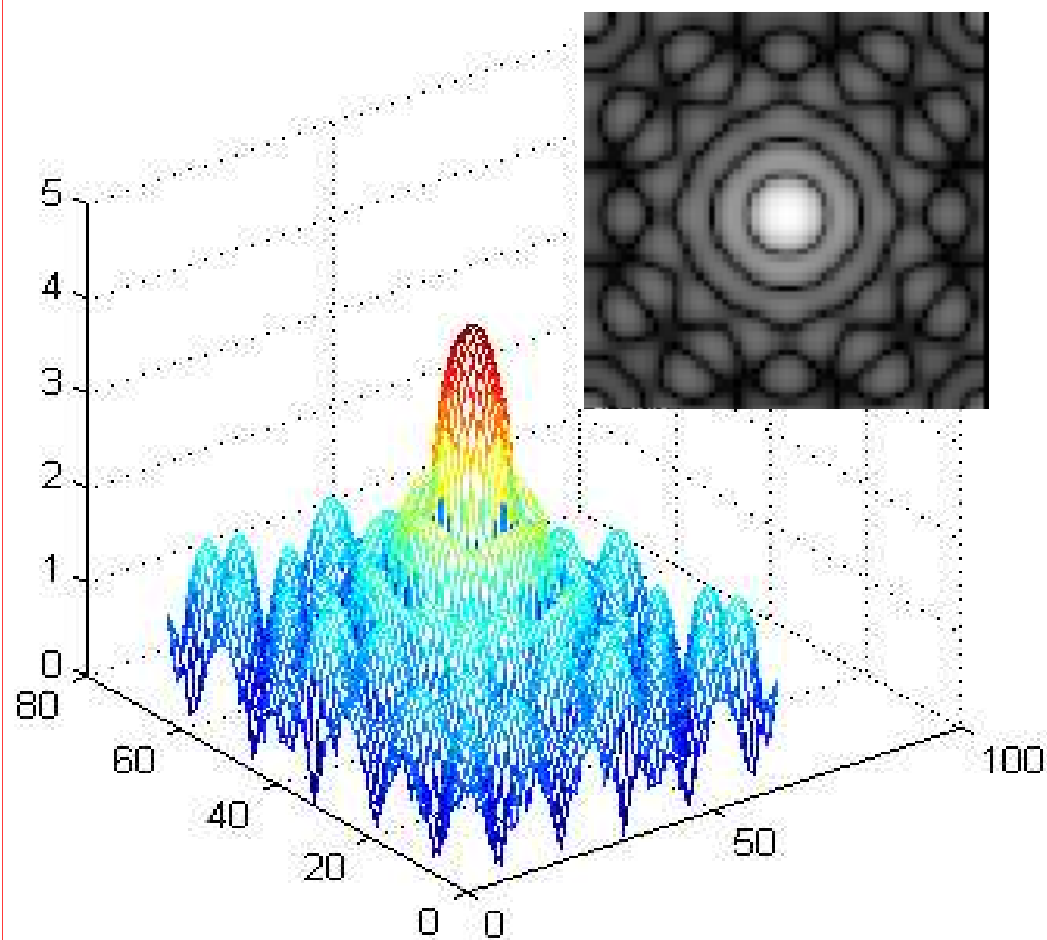


$$\mathfrak{I}^{-1}\{F(u,v)\}$$



$f(x,y)$

(64x64)



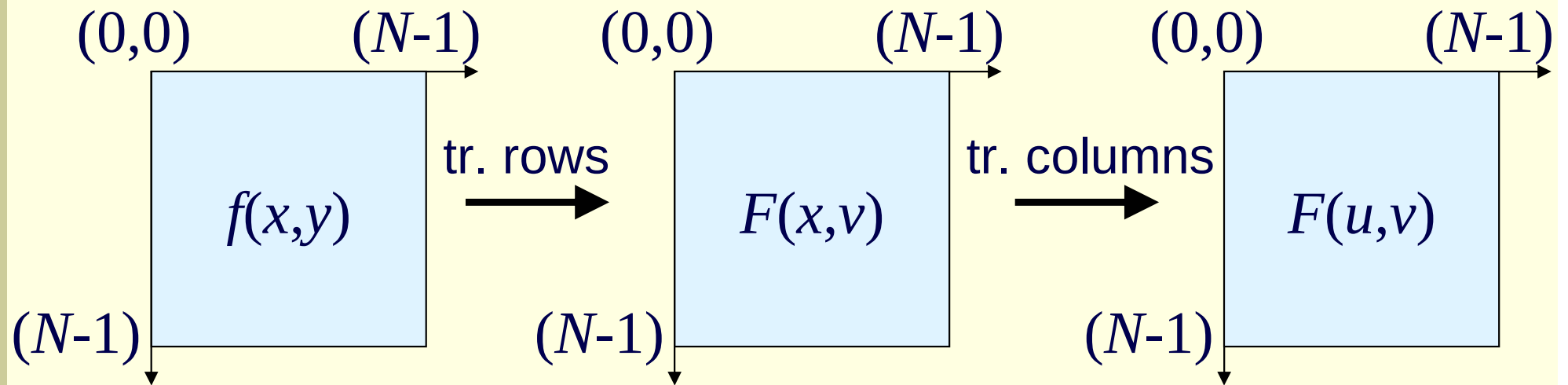
$|F(u,v)|$

$\log(1+|F(u,v)|)$

# Properties of the two-dimensional Fourier transform

---

## Separability:



Computation of the 2-D Fourier transform as a series of 1-D transforms

# Separability of the 2-D Fourier transform

---

$$F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux+vy)/N}$$

$F(x, v)$

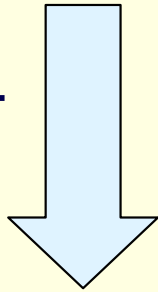
$$F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} e^{-j2\pi ux/N} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi vy/N}$$

$$F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} F(x, v) e^{-j2\pi ux/N}$$

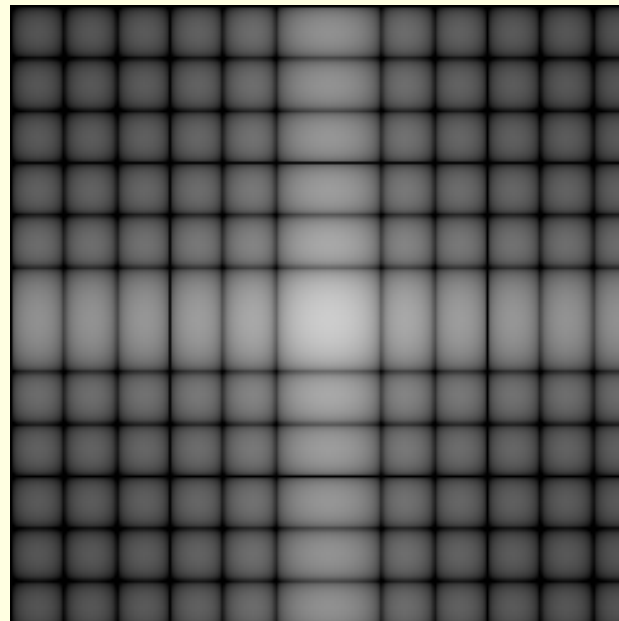
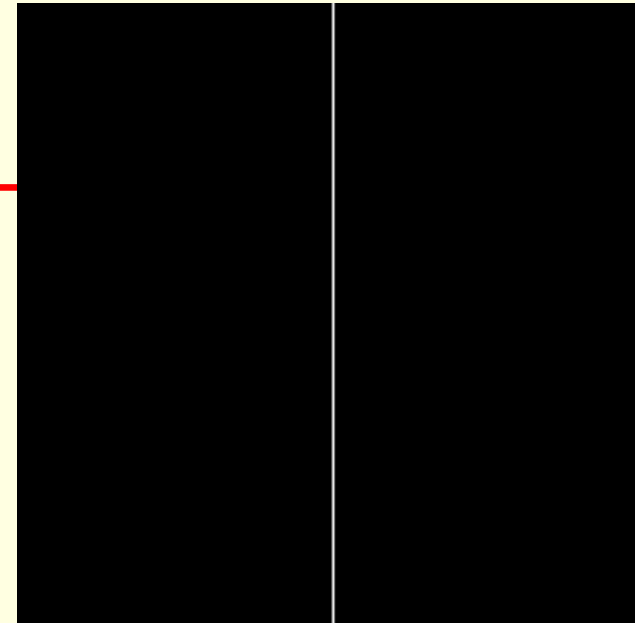
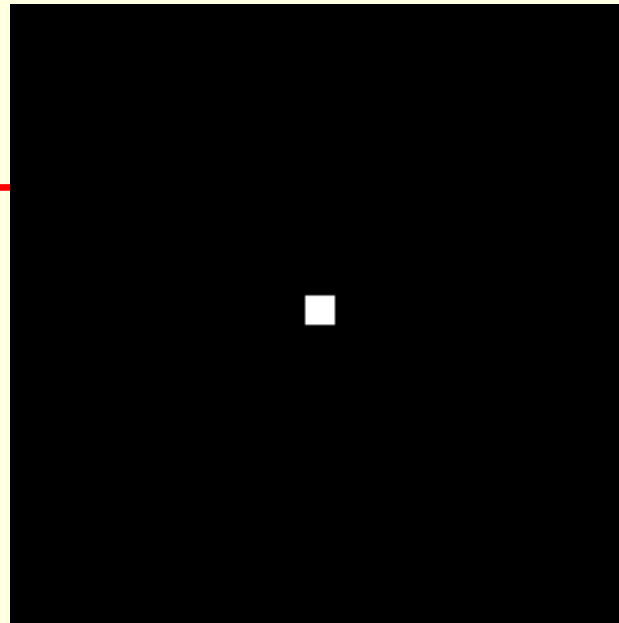
# Example

Images

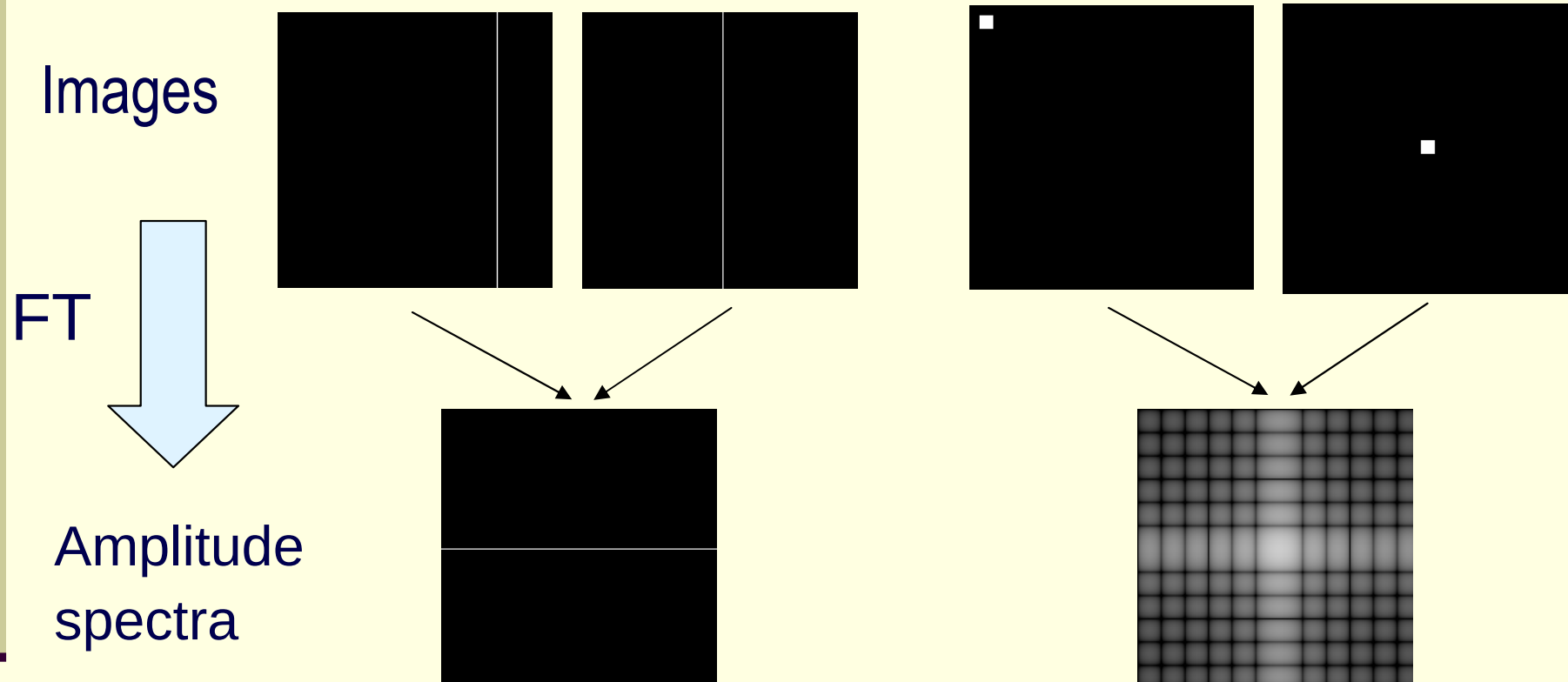
FT



Amplitude  
spectra



# Shift in the spatial domain



$$f(x - x_0, y - y_0) \Leftrightarrow F(u, v) \exp \left[ -\frac{j2\pi(ux_0 + vy_0)}{N} \right]$$

# Properties of the two-dimensional Fourier transform

---

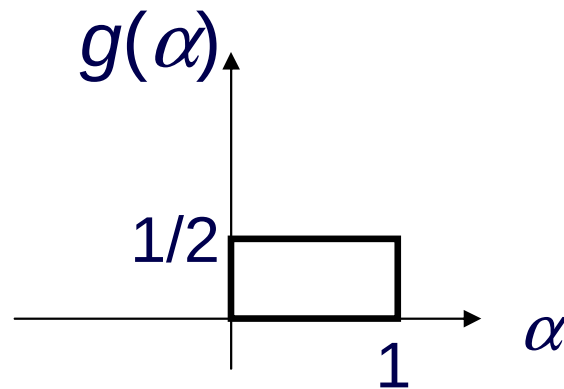
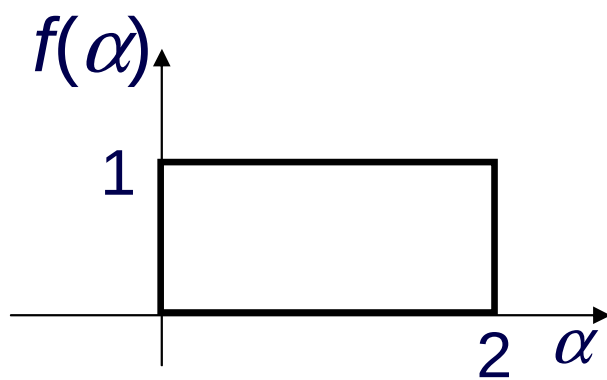
## Convolution:

$$\mathcal{F} \{f(x,y) g(x,y)\} = F(u,v) * G(u,v)$$

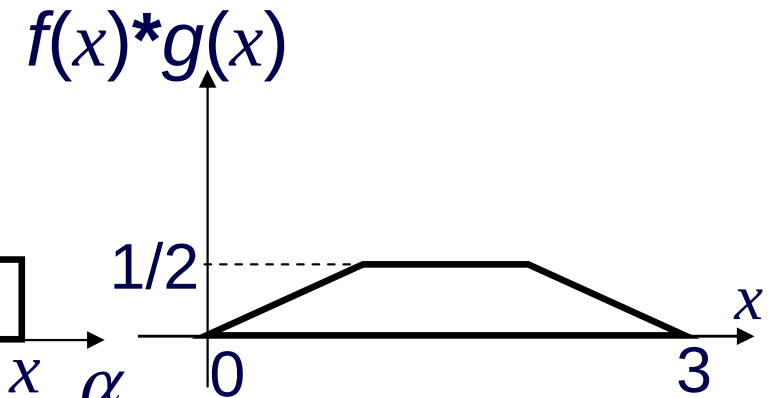
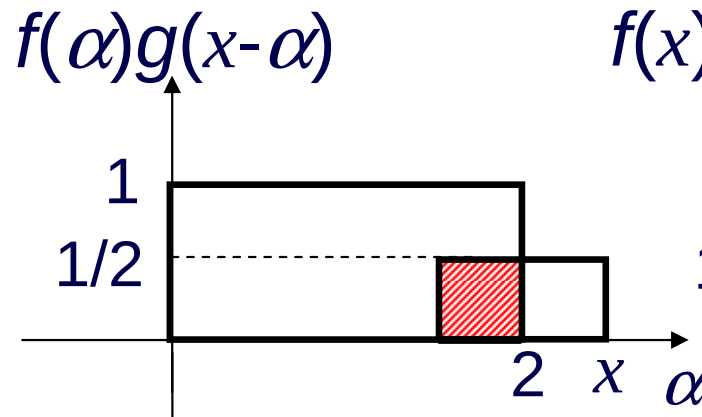
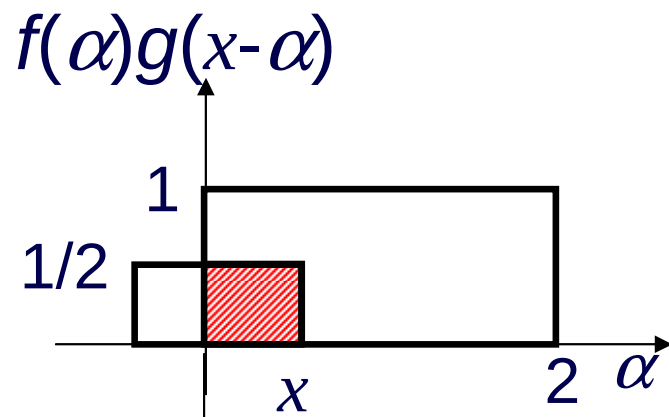
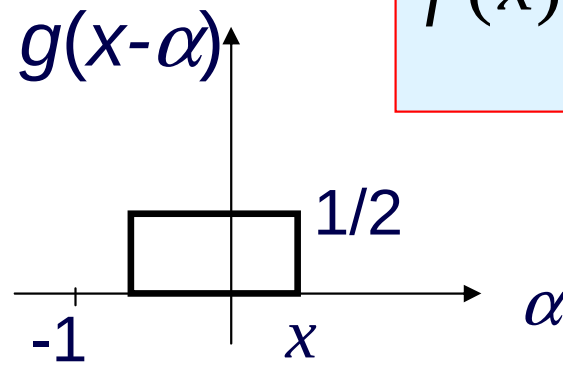
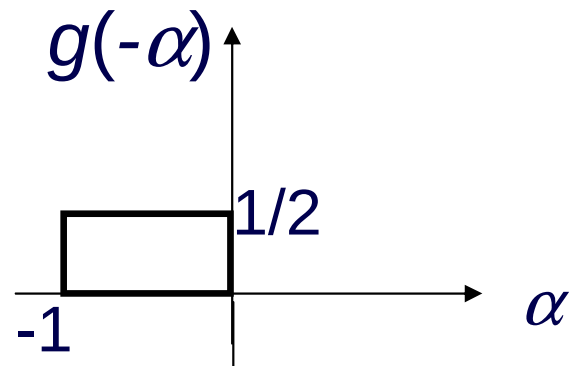
$$\mathcal{F} \{f(x,y) * g(x,y)\} = F(u,v) G(u,v)$$

This property is useful in designing digital image filters.

# 1-D convolution example



$$f(x) * g(x) = \int_{-\infty}^{\infty} f(\alpha)g(x-\alpha)d\alpha$$



## 2-D convolution of discrete functions

---

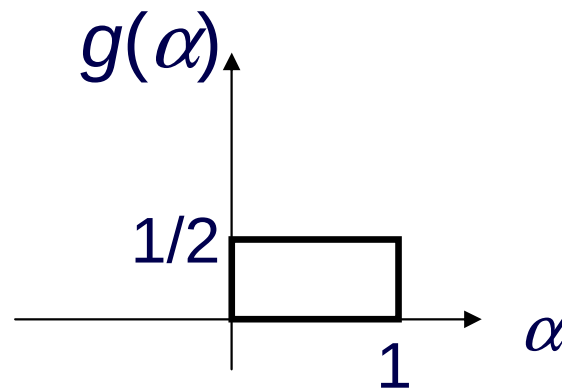
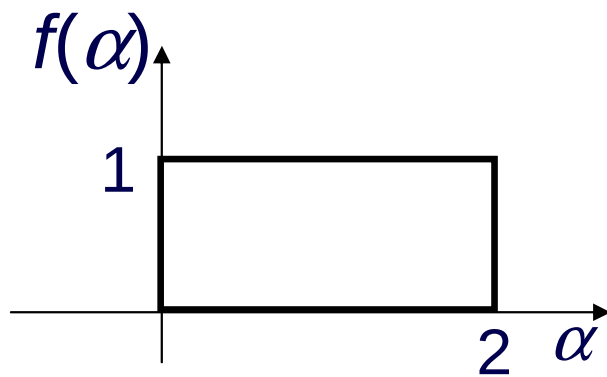
$f(i,j)$ ,  $g(i,j)$  – discrete 2-D functions of period  $N \times N$

increase periods of  $f(i,j)$  and  $g(i,j)$  up to  $M=2N-1$ :

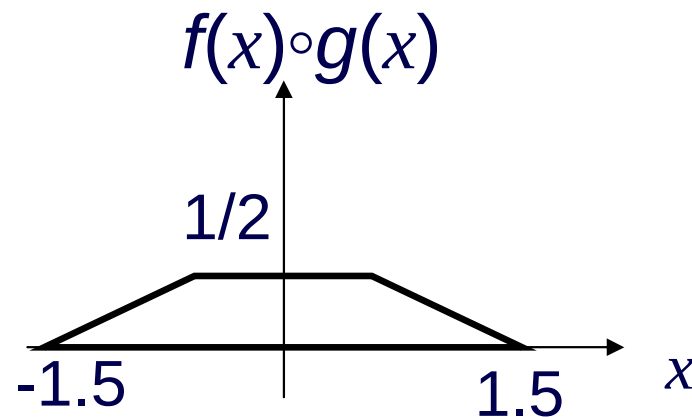
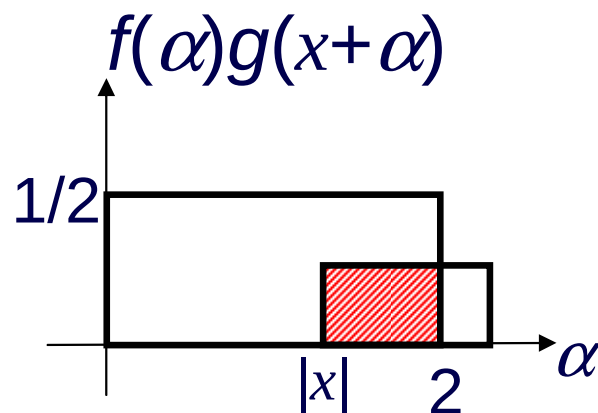
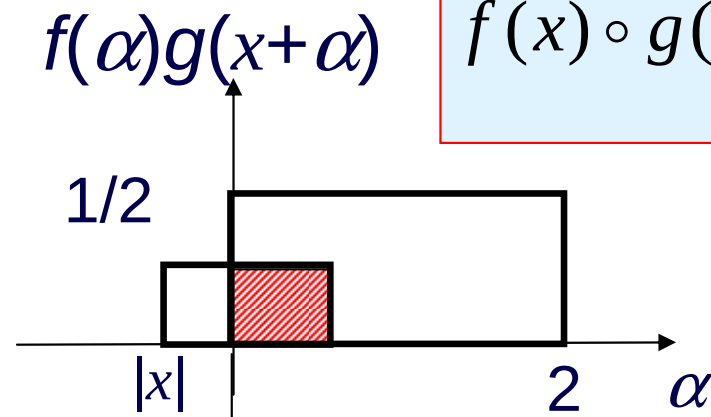
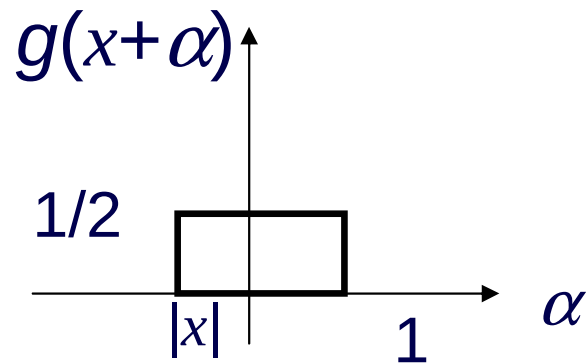
$$f_e(i,k) = \begin{cases} f(i,k) & 0 \leq i,k \leq N-1 \\ 0 & N \leq i,k \leq M-1 \end{cases} \quad g_e(i,k) = \begin{cases} g(i,k) & 0 \leq i,k \leq N-1 \\ 0 & N \leq i,k \leq M-1 \end{cases}$$

$$f_e(i,k) * g_e(i,k) = \sum_{m=0}^{M-1} \sum_{n=0}^{M-1} f_e(m,n) g_e(i-m, k-n)$$

# 1-D correlation - example



$$f(x) \circ g(x) = \int_{-\infty}^{\infty} f(\alpha)g(x+\alpha)d\alpha$$



# Correlation of 2-D discrete functions

---

$f(i,j)$ ,  $g(i,j)$  – discrete 2-D functions of period  $N \times N$

Increase the periods as for convolution:

$$f_e(i,k) \circ g_e(i,k) = \sum_{m=0}^{M-1} \sum_{n=0}^{M-1} f_e(m,n) g_e(i+m, k+n)$$

# Periodicity of the FT

---

$$F(u,v) = F(u+N,v) = F(u,v+N) = F(u+N,v+N)$$

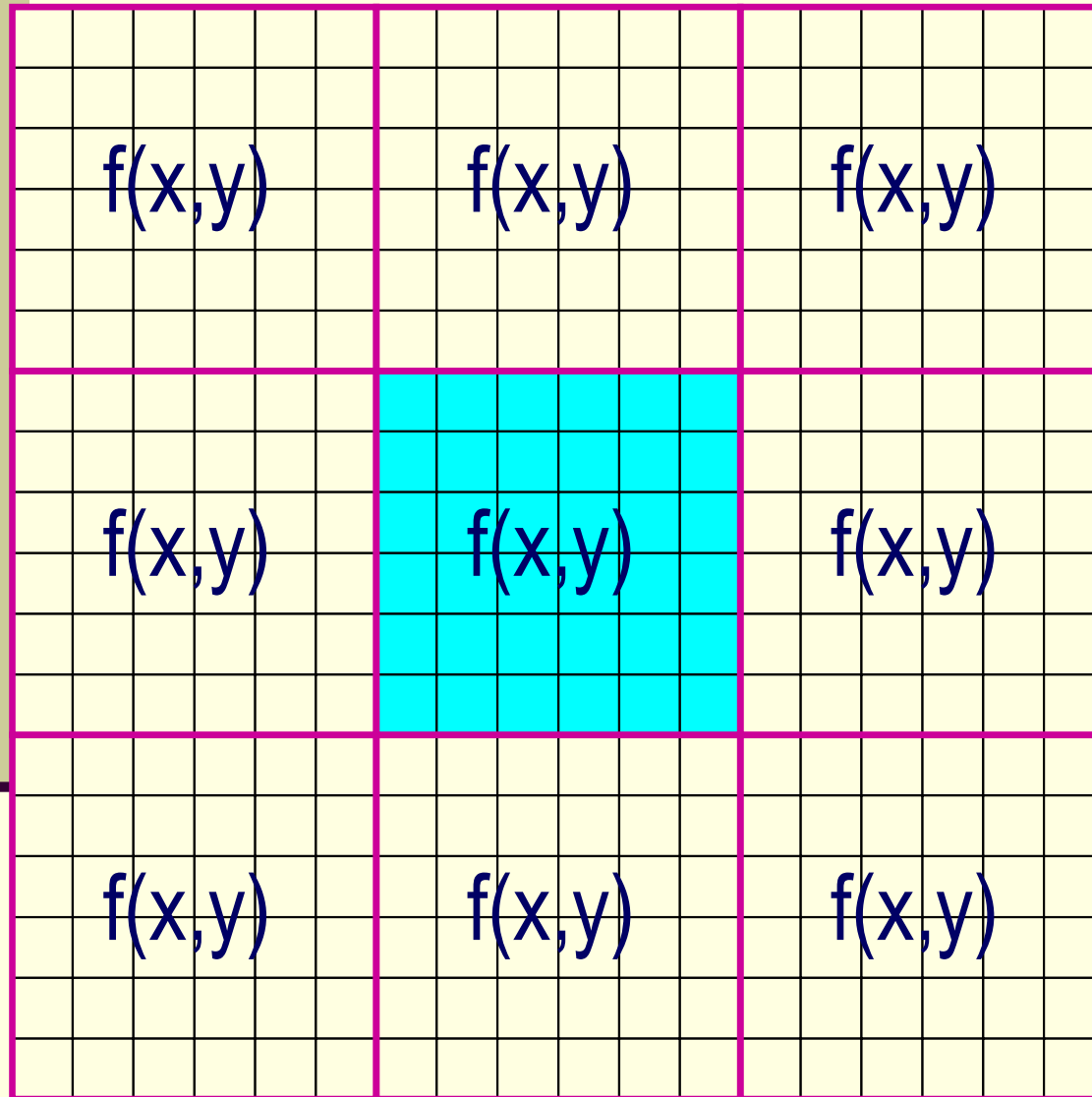
If  $f(x,y)$  is a real valued function then:

$$F(u,v) = F^*(-u,-v)$$

and:

$$|F(u,v)| = |F(-u,-v)|$$

# Fourier transform of images



*It is assumed the transformed image is a periodic function of period  $(N, N)$*

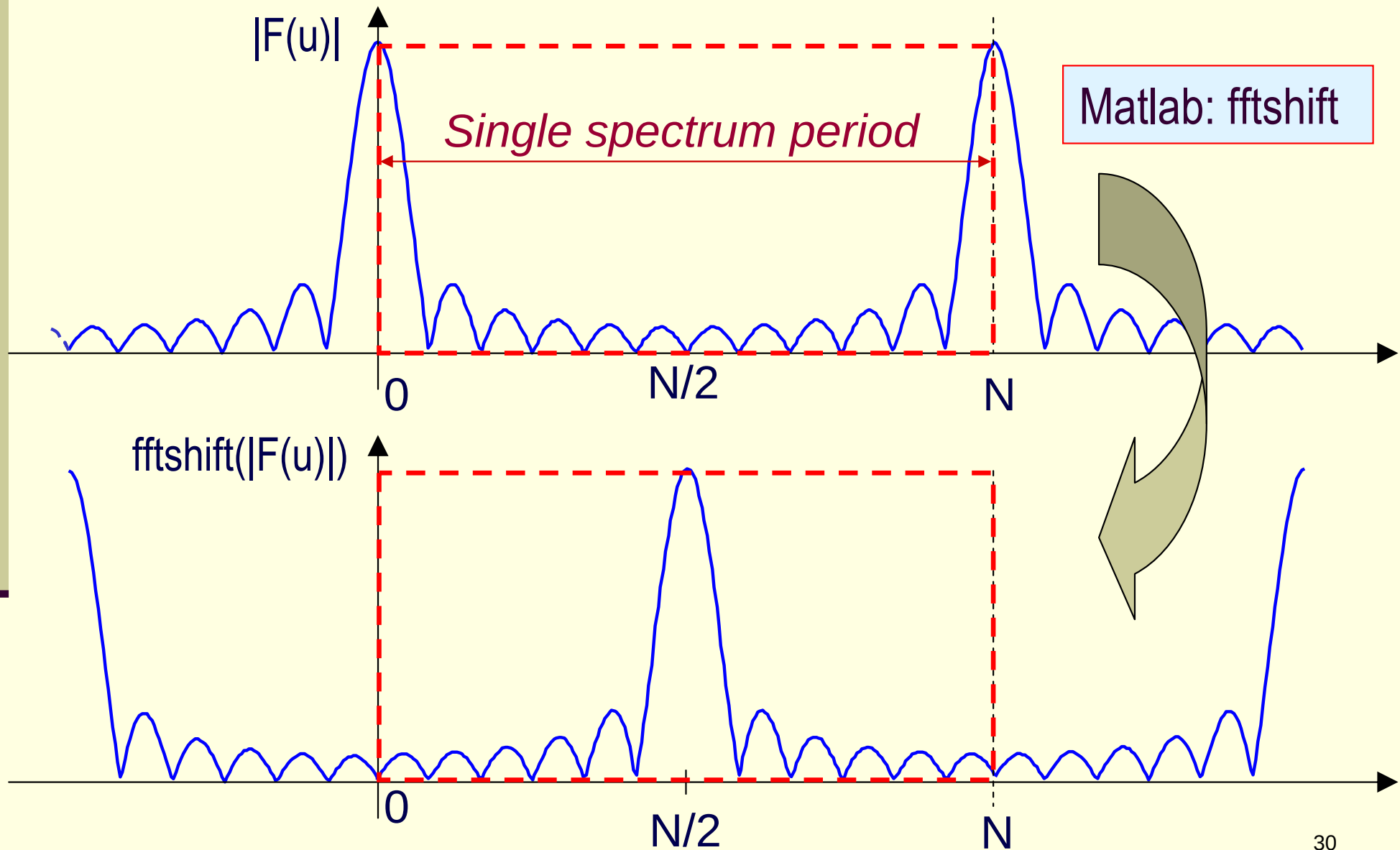
# Translation in the spectral domain

---

$$f(x, y) \exp\left[\frac{j2\pi(u_0x + v_0y)}{N}\right] \Leftrightarrow F(u - u_0, v - v_0)$$

This Fourier property is known as the theorem of modulation.

# Translation in the spectral domain



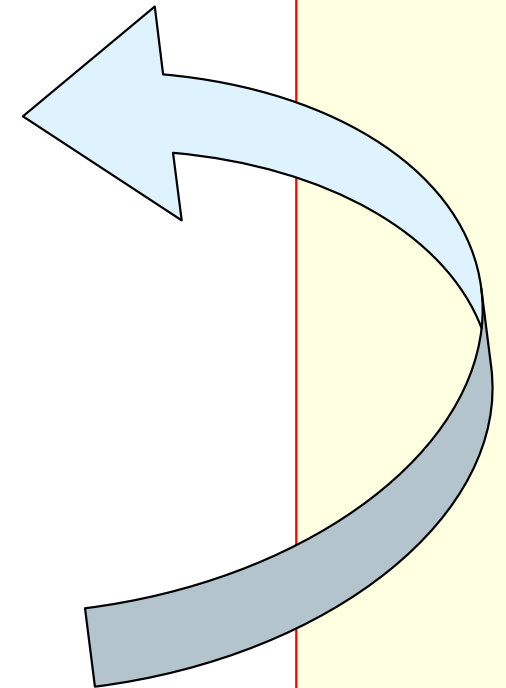
# Translation in the spectral domain

---

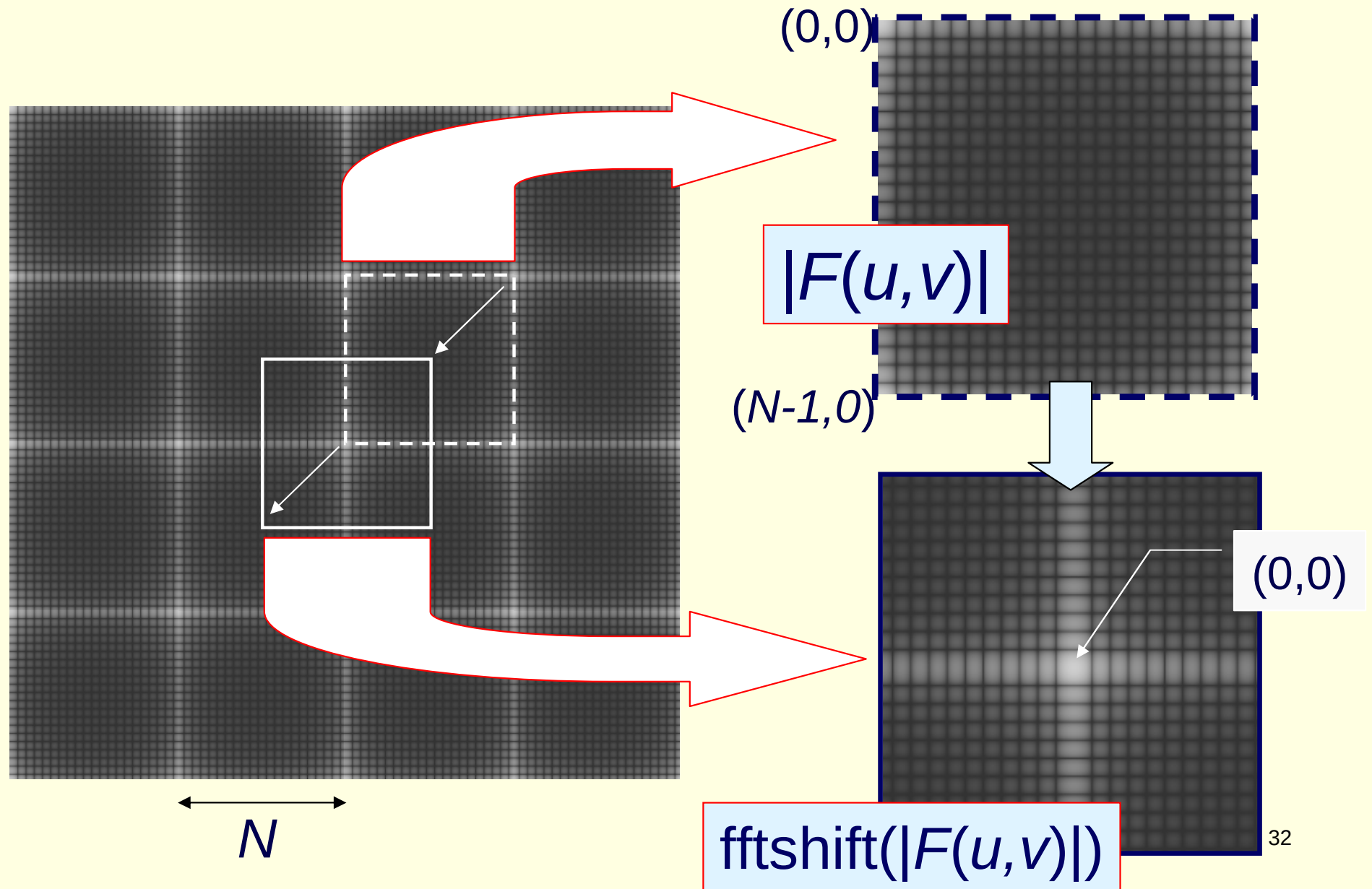
$$f(x, y) \exp\left[\frac{j2\pi(u_0 x + v_0 y)}{N}\right] \Leftrightarrow F(u - u_0, v - v_0)$$

$$\text{for } u_0 = v_0 = \frac{N}{2} \quad \Leftrightarrow F\left(u - \frac{N}{2}, v - \frac{N}{2}\right)$$

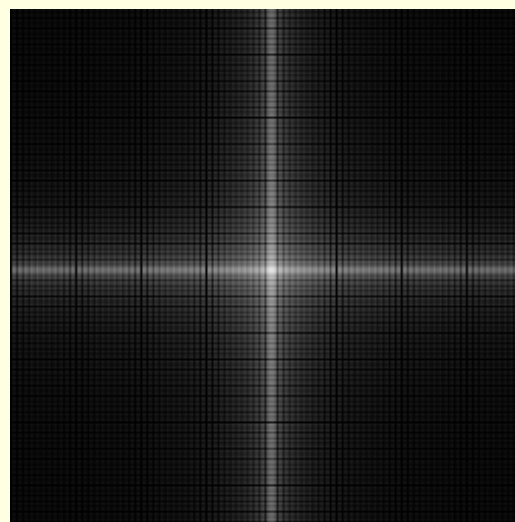
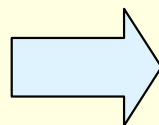
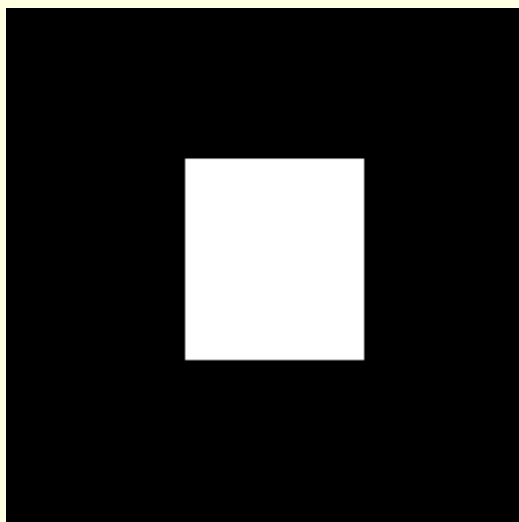
$$\begin{aligned} f(x, y) \exp\left[\frac{j2\pi(u_0 x + v_0 y)}{N}\right] &= \\ &= f(x, y) \exp[j\pi(x + y)] = f(x, y)(-1)^{x+y} \end{aligned}$$



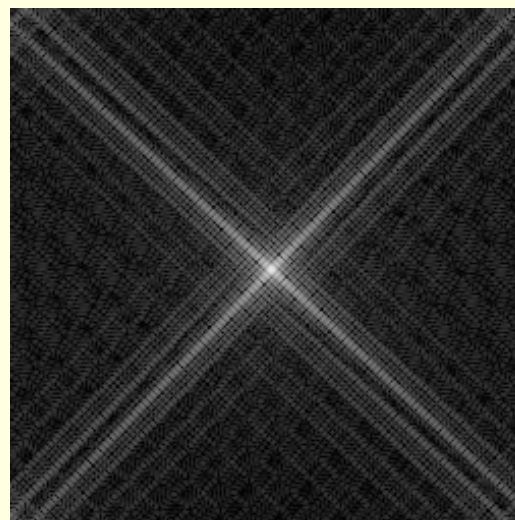
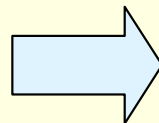
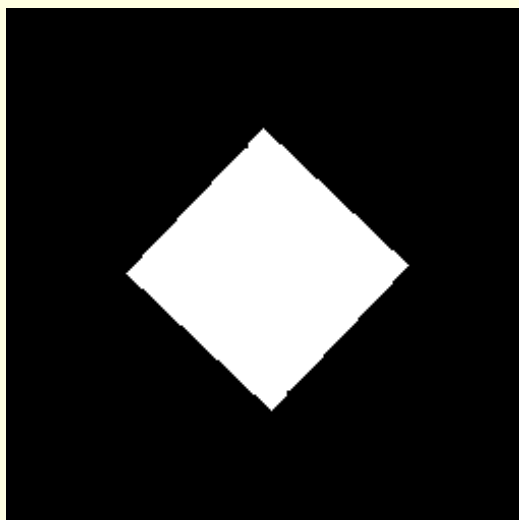
# Translation in spectral domain



# Rotation



$$\theta_0 = 0^\circ$$

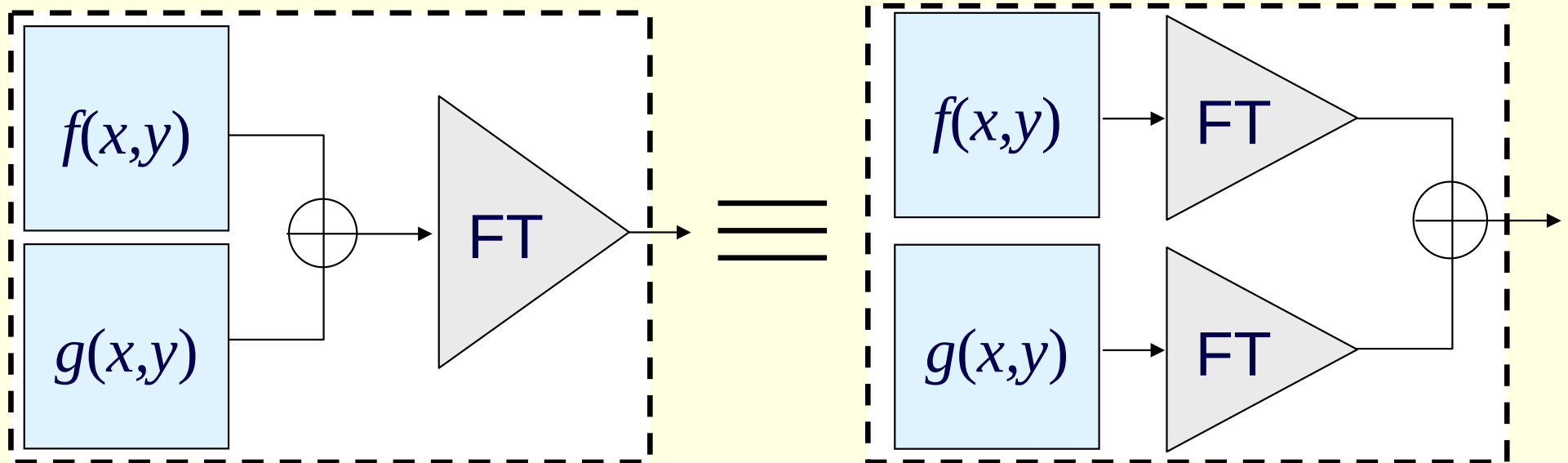


$$\theta_0 = 45^\circ$$

$$f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \phi + \theta_0)$$

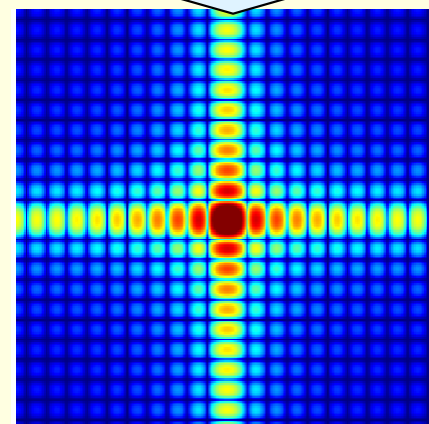
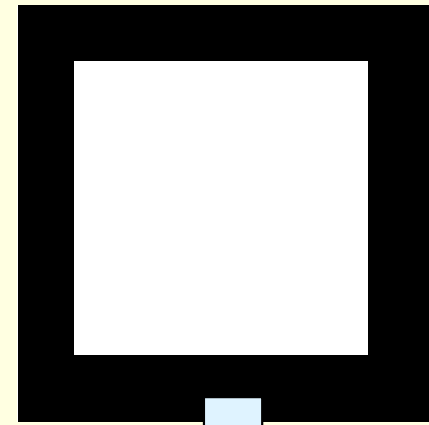
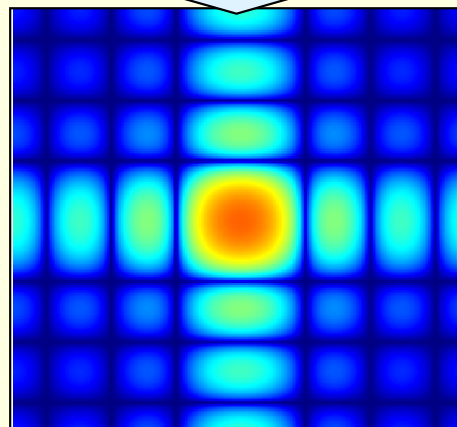
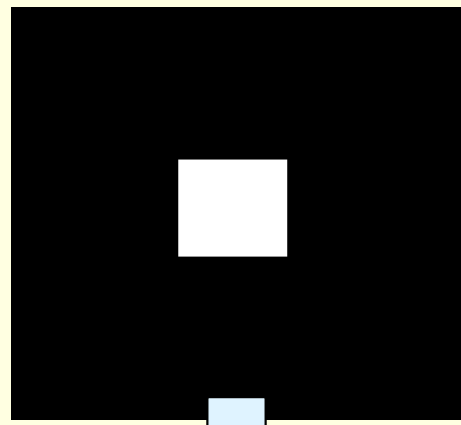
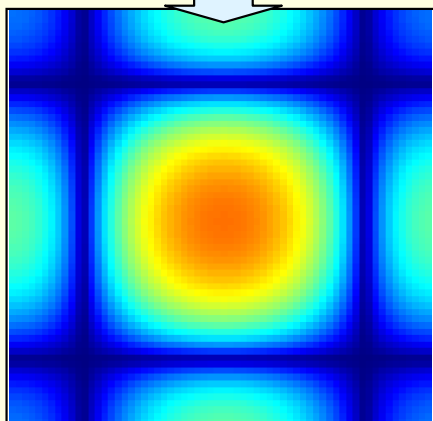
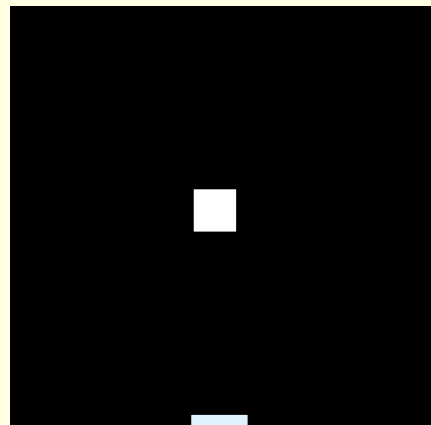
# Linearity

$$\mathcal{F}\{a f(x,y) + b g(x,y)\} = a F(u,v) + b G(u,v)$$



# Scaling

$$\mathcal{F}\{f(ax,by)\} = |ab|^{-1} F(u/a, v/b) \quad a,b \in \mathbb{R}$$

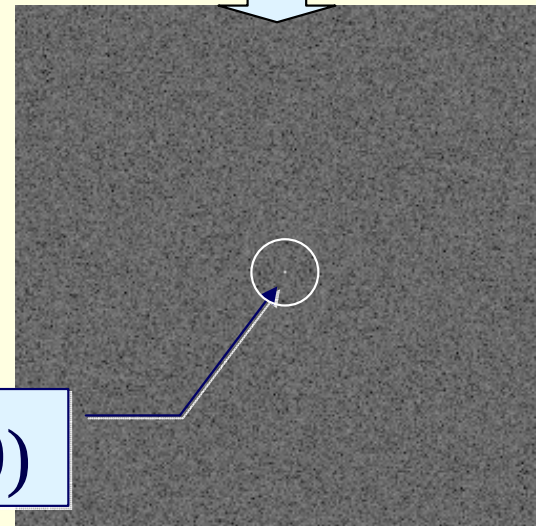
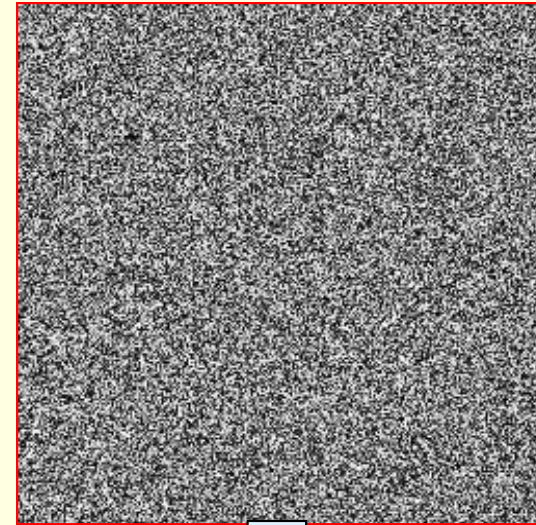


# Average value

$$\bar{f}(x,y) = \frac{1}{N^2} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y)$$

$$F(0,0) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y)$$

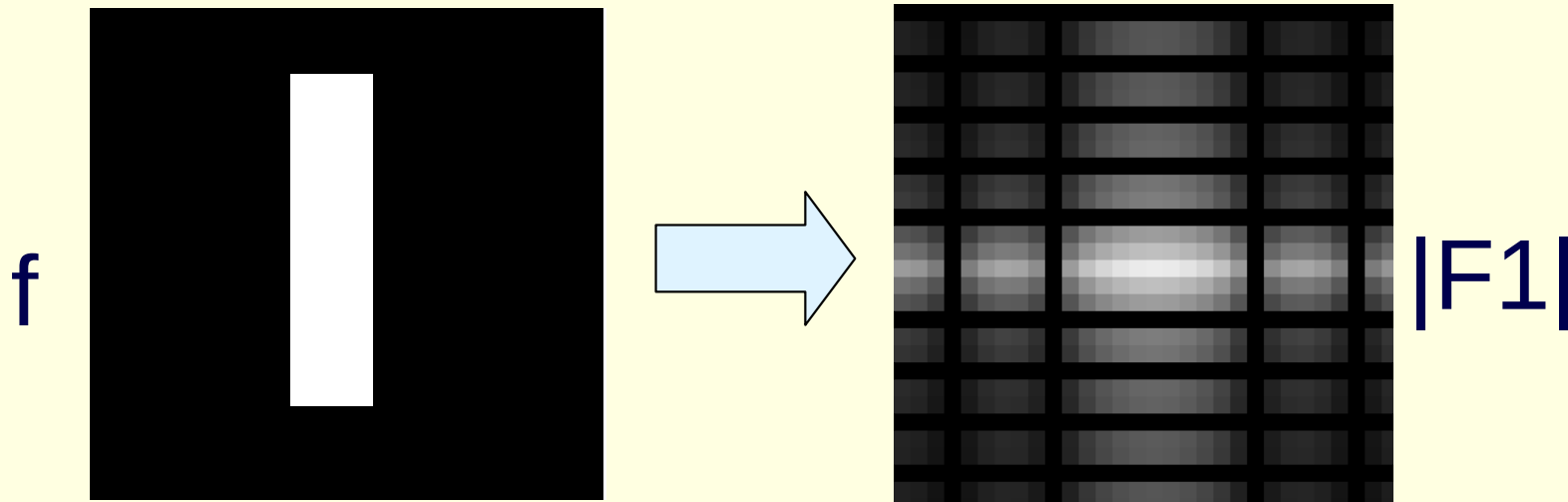
$$\bar{f}(x,y) = \frac{1}{N} F(0,0)$$



$F(0,0)$

# Fourier transform of an image - examples

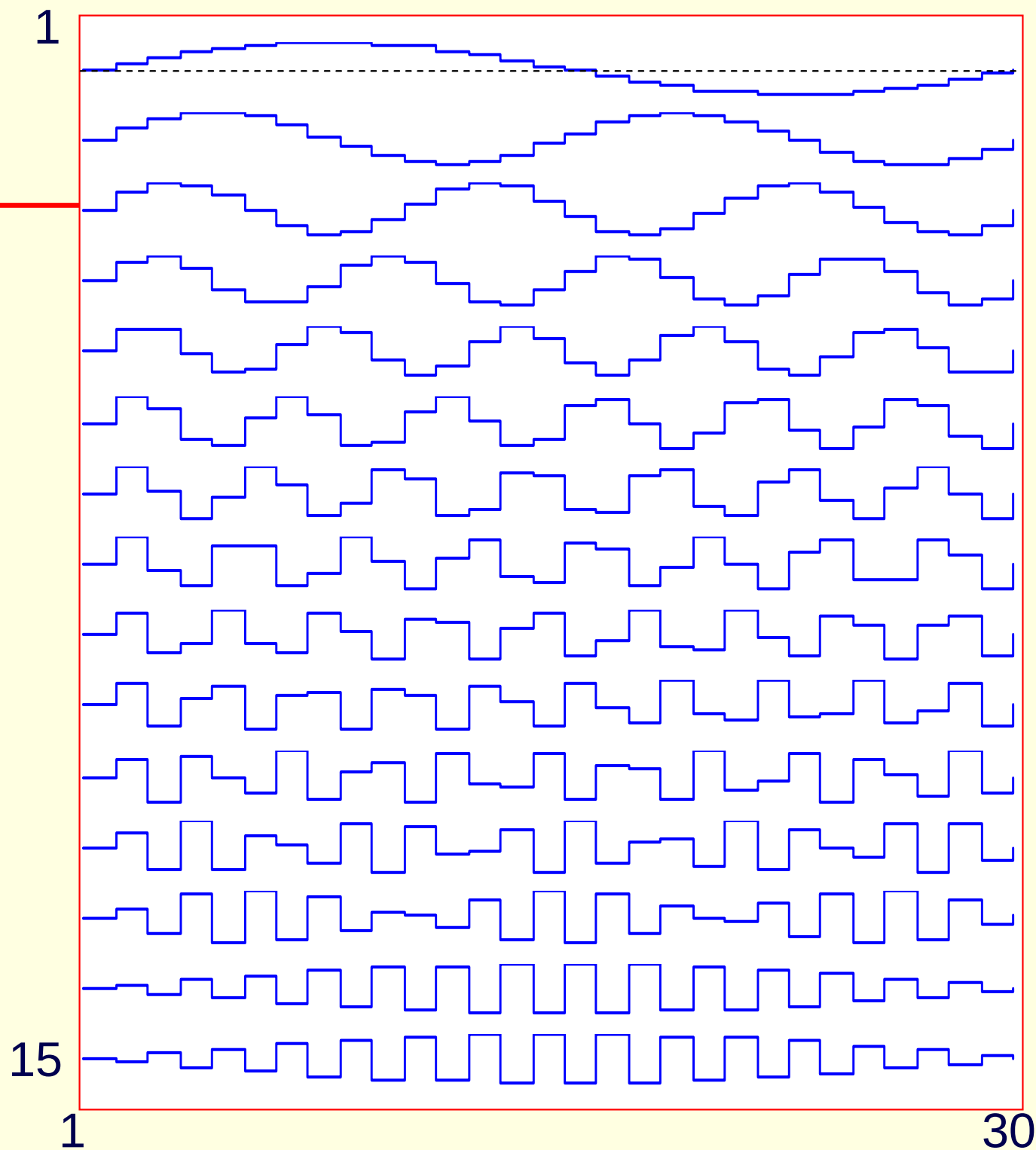
---



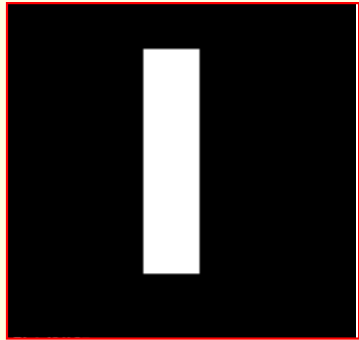
```
f=zeros(30,30);  
f(5:24,13:17)=1;  
  
imshow(f,'notruesize')  
F=fft2(f);           %compute 2-D Fourier transform  
F1=log(abs(F)+1);    %amplitude spectrum  
imshow(F1,[0 5], 'notruesize');
```

# Discrete Fourier Transform

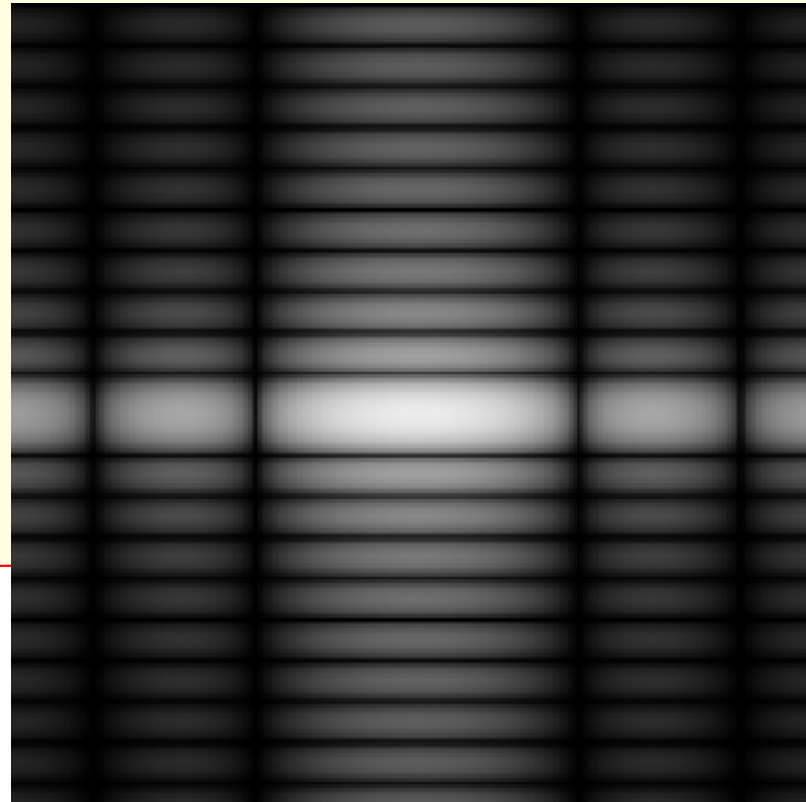
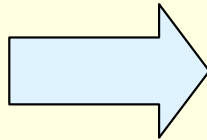
Basis functions  
for 30-point  
Fourier  
transform (sine  
component)



# Fourier transform of an image - examples



$f$



$|F2|$

**%better resolution**

```
f=zeros(30,30);  
f(5:24,13:17)=1;  
F=fft2(f,256,256);  
F2=log(abs(F)+1);  
imshow(fftshift(F2),[0 5],'notruesize');
```

# The Fast Fourier Transform, FFT (*successive doubling method*)

---

If  $N=2^n$ , then  $N=2*M$  and one can show that:

$$F_{\text{even}}(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(2x) W_M^{ux}, \quad F_{\text{odd}}(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(2x+1) W_M^{ux}$$

$$F(u) = \frac{1}{2} [F_{\text{even}}(u) + F_{\text{odd}}(u) W_{2M}^u], \quad u = 0, 1, \dots, M-1$$

$$F(u+M) = \frac{1}{2} [F_{\text{even}}(u) - F_{\text{odd}}(u) W_{2M}^u], \quad u = 0, 1, \dots, M-1$$

$$W_M = e^{-j2\pi/M}$$

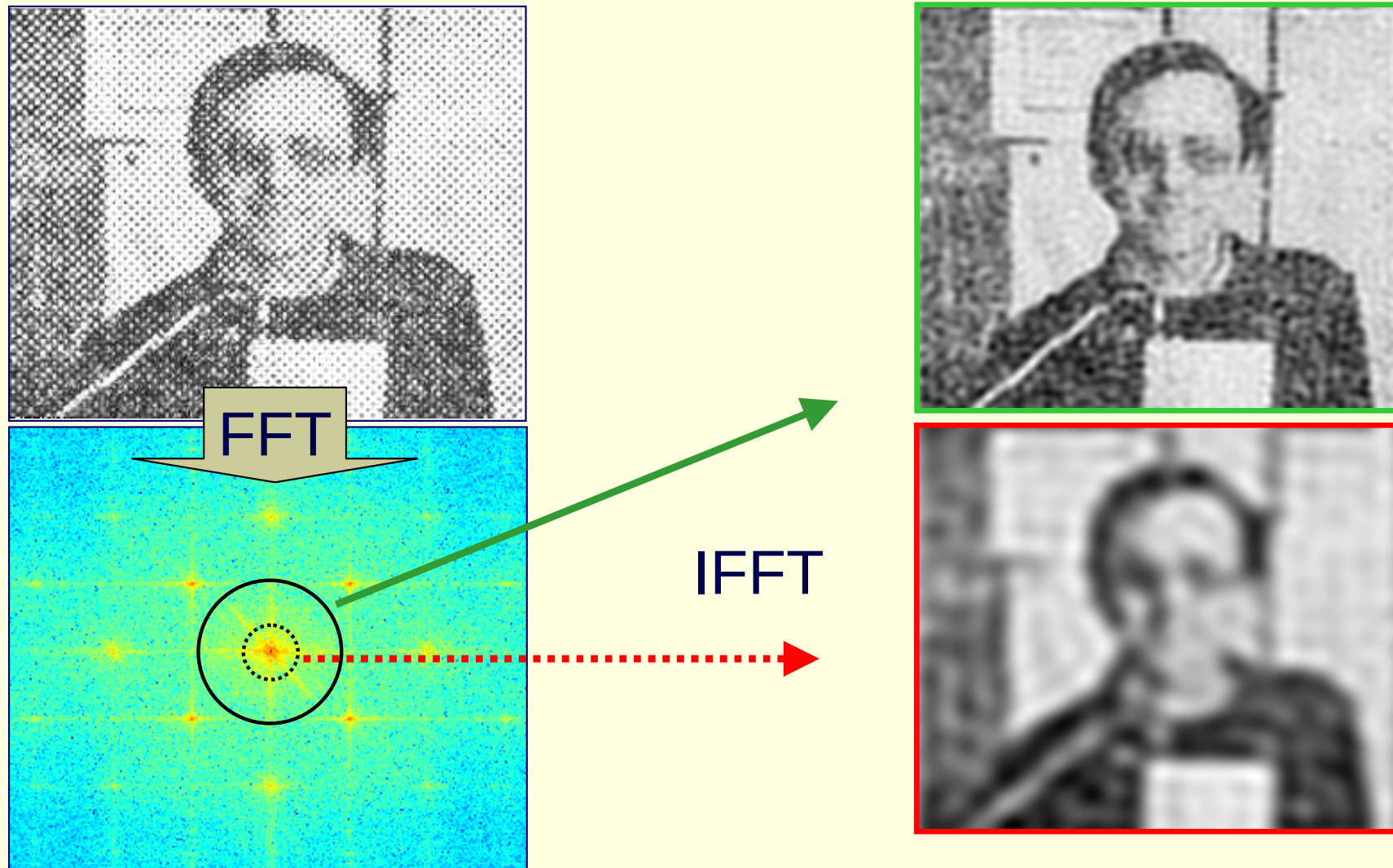
# Comparison of TF and FFT

---

| N    | $N^2$ (FT) | $N \log N$ (FFT) | Advantage<br>$N / \log N$ |
|------|------------|------------------|---------------------------|
| 16   | 256        | 64               | 4                         |
| 256  | 65535      | 2048             | 32                        |
| 512  | 262144     | 4608             | 64                        |
| 2048 | $\sim 4e6$ | 22528            | 186                       |

# 2-D Fourier transform

## Interactive noise reduction in Fourier spectrum

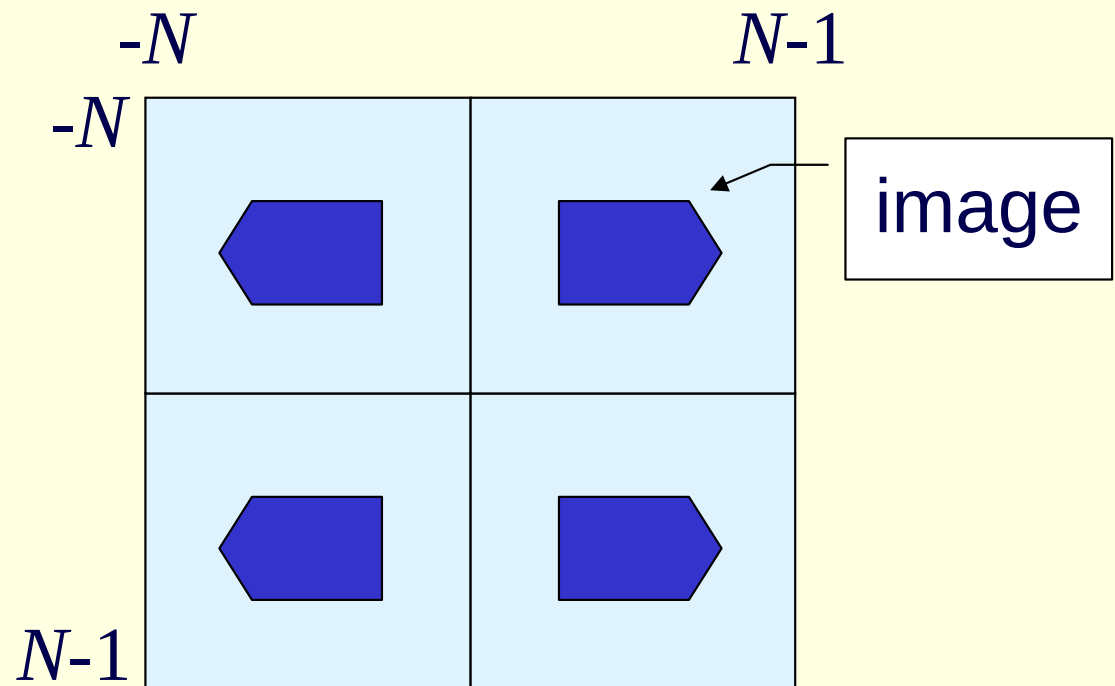


# Discrete Cosine Transform (DCT)

$$F(u,v) = \frac{2}{N} \sum_{i=0}^{N-1} \sum_{k=0}^{N-1} f(x,y) \cos\left[\frac{\pi u(2x+1)}{2N}\right] \cos\left[\frac{\pi v(2y+1)}{2N}\right]$$

for:  $u, v = 1, 2, \dots, N-1$

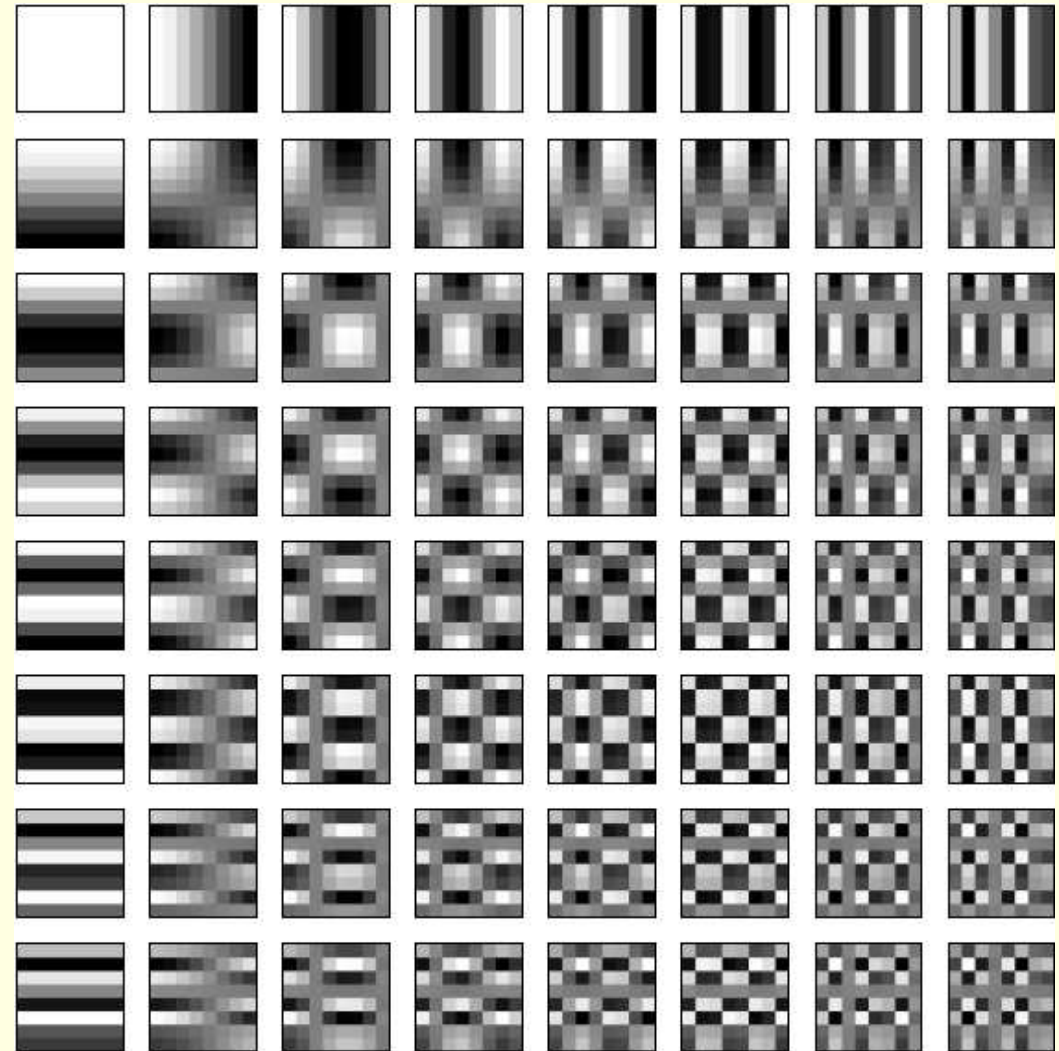
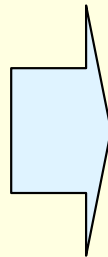
Fourier spectrum of a real valued and symmetric function has real valued coefficients, ie. only those associated with the cosine components of the Fourier series



# DCT basis functions

---

DCT basis  
functions for 8x8  
image blocks



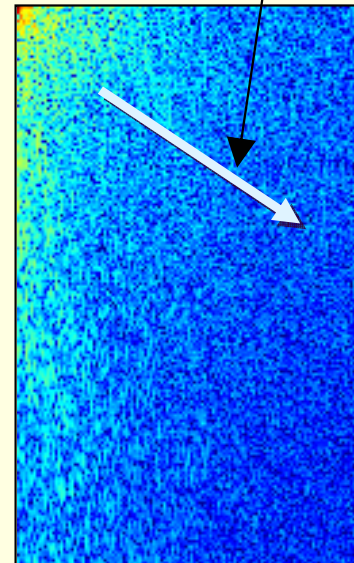
# Discrete Cosine Transform (DCT)

---



‘autumn’ image

(0,0)



fast vanishing of  
the coefficients

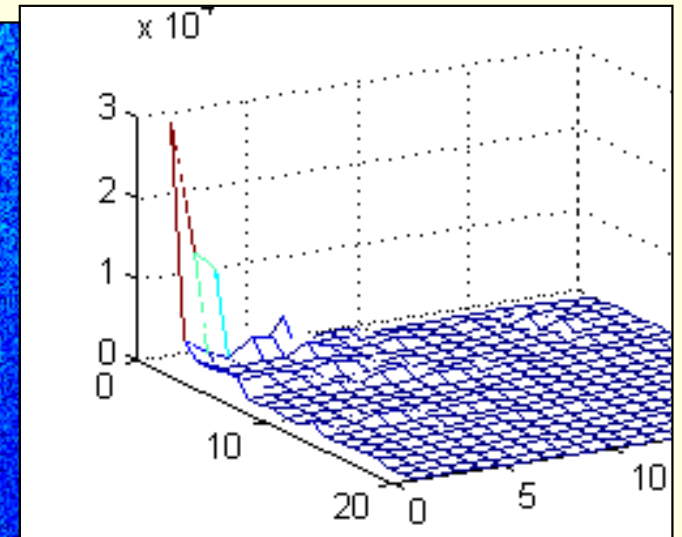
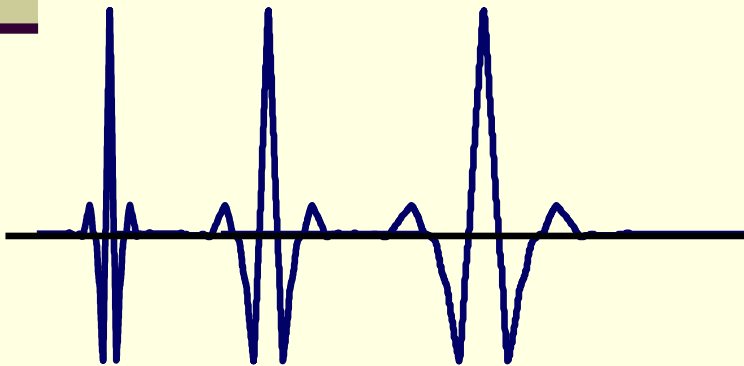


image cosine transform

**The JPEG image compression standard  
is based on DCT**

# Other image transforms

- the **Karhunen-Loeve** transform - equivalent to the **PCA** (*Principal Component Analysis*)
- the wavelet transform is used in JPEG-2000 image coding standard



eigenfaces



©AT&T Labs



# Other image transforms

---

