

# Summary of last week's lecture

- Virtually all filtering is a local neighbourhood operation
- Convolution = linear and shift-invariant filters
  - e.g. mean filter, Gaussian weighted filter
  - kernel can sometimes be decomposed
- Many non-linear filters exist also
  - e.g. median filter, bilateral filter
- The Fourier transform decomposes a function (image) into trigonometric basis functions (sines & cosines).
- The Fourier transform is used to analyse frequency components of an image.



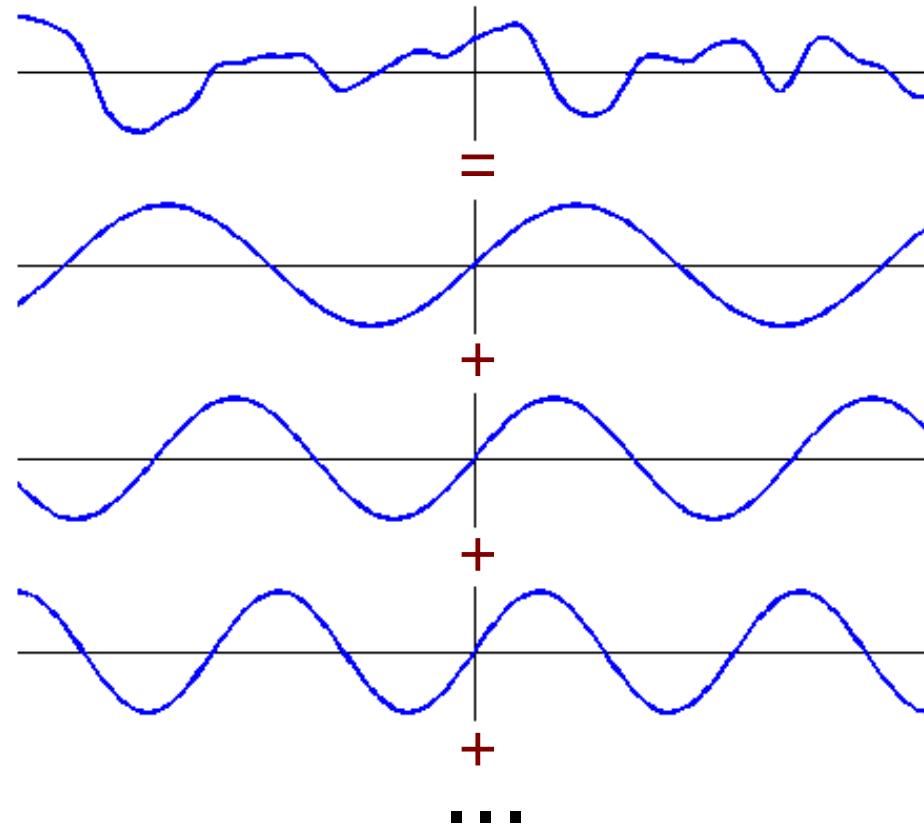
# Convolution properties

- Linear:
  - Scaling invariant:  $(C f) \otimes h = C(f \otimes h)$
  - Distributive:  $(f+g) \otimes h = f \otimes h + g \otimes h$
- Time Invariant:  
 $(= shift\ invariant)$   $shift(f) \otimes h = shift(f \otimes h)$
- Commutative:  $f \otimes h = h \otimes f$
- Associative:  $f \otimes (h_1 \otimes h_2) = (f \otimes h_1) \otimes h_2$

# Fourier transform

$$\mathcal{F}(\omega) = \int_{-\infty}^{\infty} f(x) e^{-i\omega x} dx$$

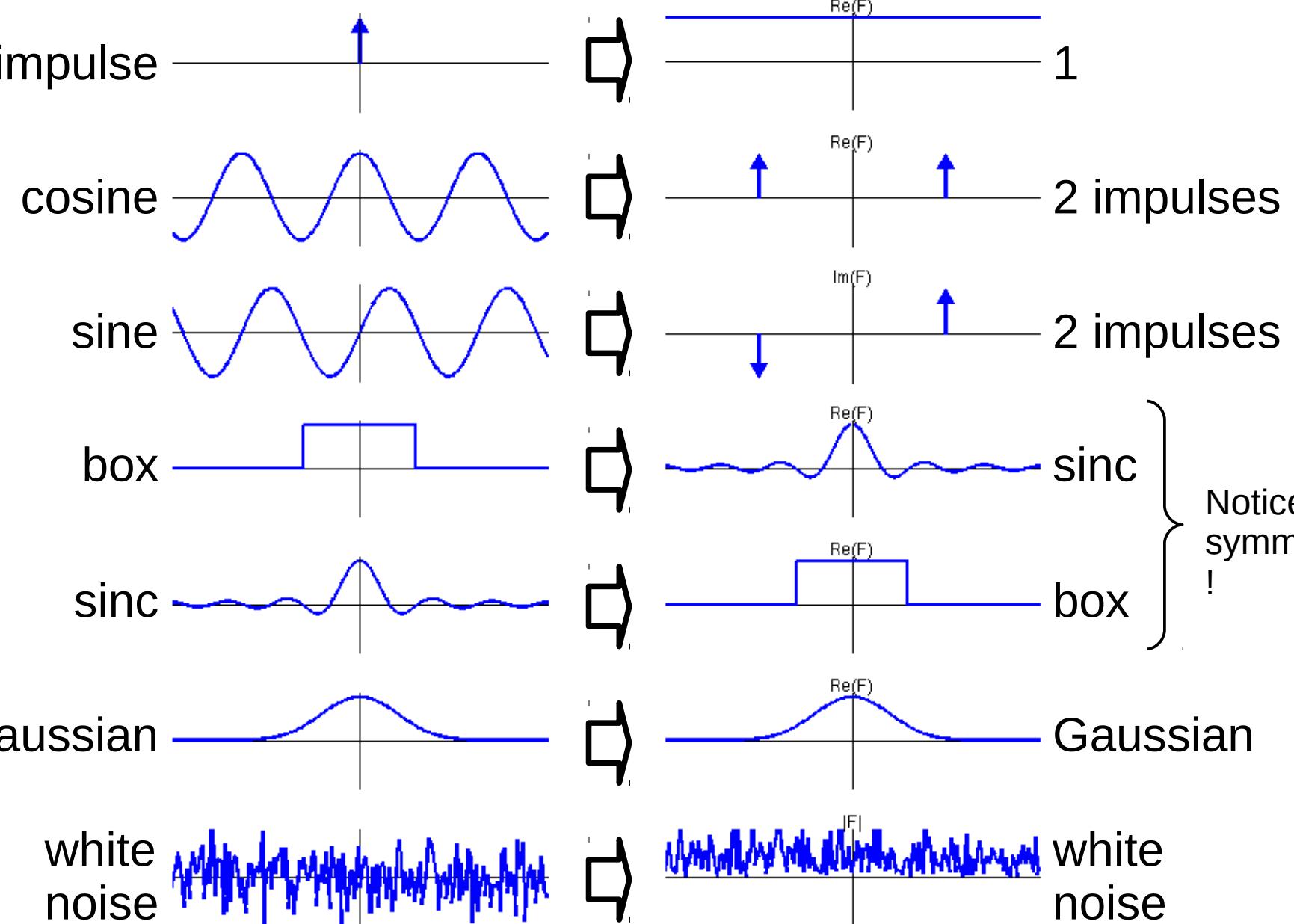
$$\begin{aligned} f(x) &= \\ &= \mathcal{F}(\omega_1) e^{i\omega_1 x} \\ &+ \mathcal{F}(\omega_2) e^{i\omega_2 x} \\ &+ \mathcal{F}(\omega_3) e^{i\omega_3 x} \\ &+ \dots \end{aligned}$$



$$\mathcal{F}(-\omega_1) = \mathcal{F}^*(\omega_1)$$

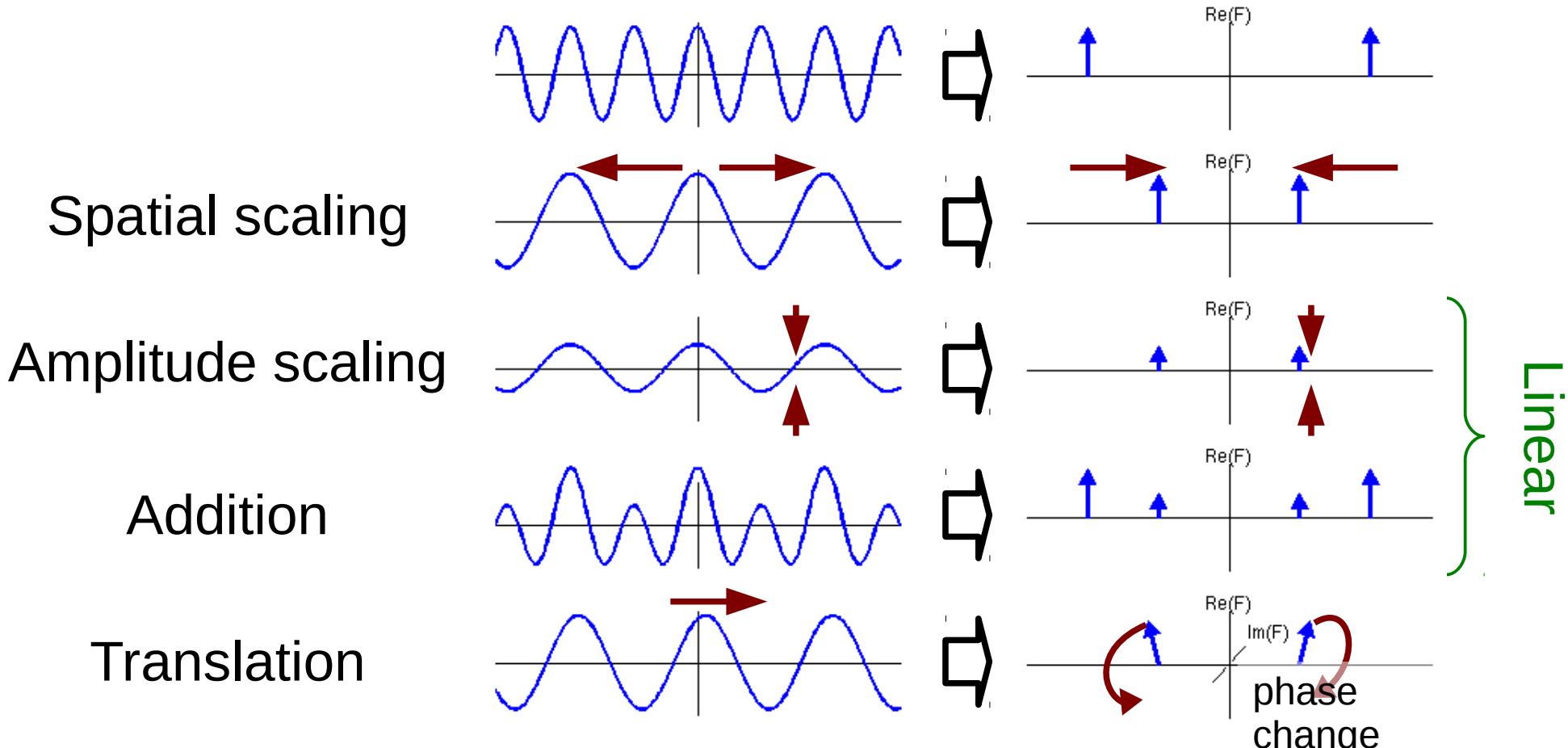
# Fourier transform pairs

Spatial



Frequency

# Properties of the Fourier transform



$$\mathcal{F}\{f \otimes h\} = \mathcal{F}\{f\} \cdot \mathcal{F}\{h\}$$

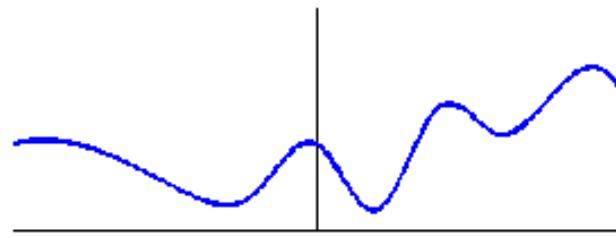
$$\mathcal{F}\{f\} \cdot \mathcal{F}\{h\} = \mathcal{F}\{f \otimes h\}$$

# Today's lecture

- The Discrete Fourier transform (DFT)
- The Fourier transform in 2D
- The Fast Fourier Transform (FFT) algorithm
- Designing filters in the Fourier domain
  - filtering out structured noise
- Sampling, aliasing, interpolation

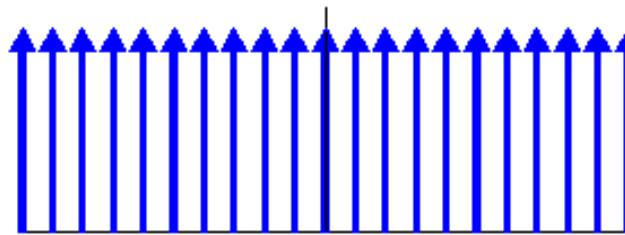
# Sampling

continuous  
function

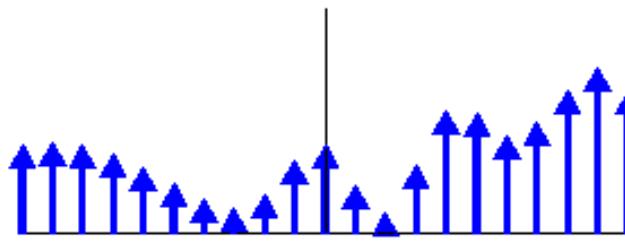


spatial domain

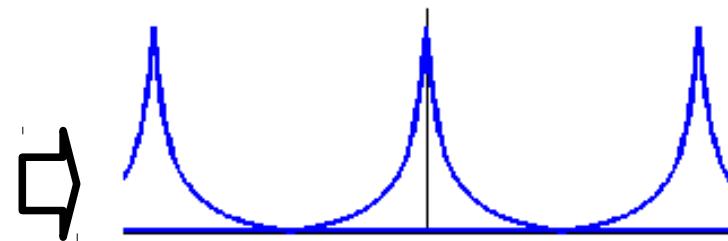
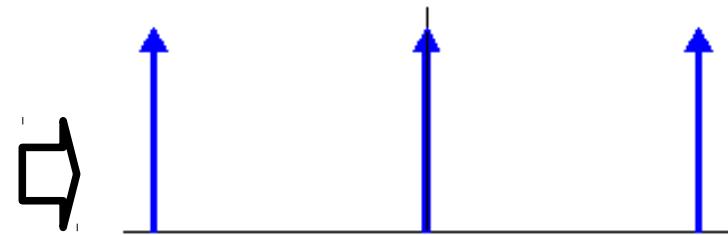
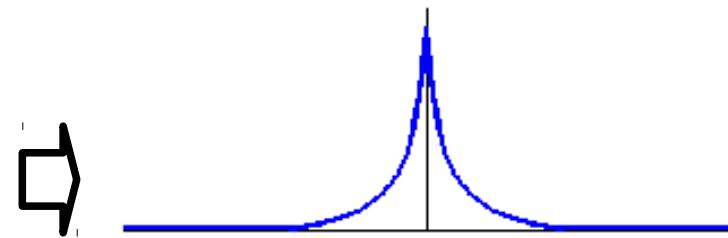
sampling  
function



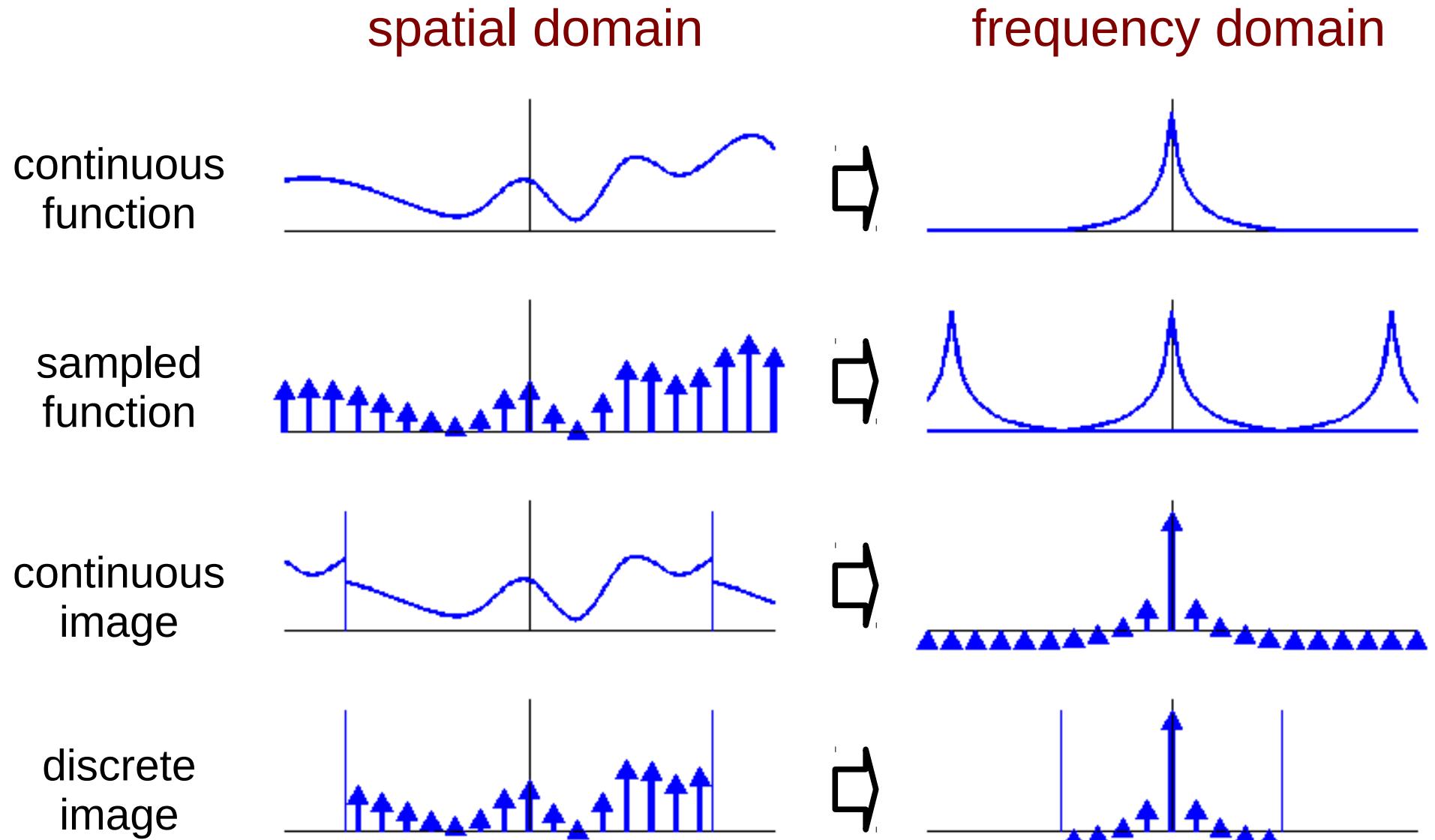
sampled  
function



frequency domain



# Discrete Fourier transform



# Discrete Fourier transform

Continuous FT:  $F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-i\omega x} dx$

Discrete FT:  $F[k] = \sum_{n=0}^{N-1} f[n] e^{-i \frac{2\pi}{N} kn}$

$k$  is the spatial frequency,  $k \in [0, N-1]$

$$\omega = 2\pi k / N$$

$$\omega \in [0, 2\pi[$$

# Discrete Fourier transform

$$F[k] = \sum_{n=0}^{N-1} f[n] e^{-i \frac{2\pi}{N} kn}$$

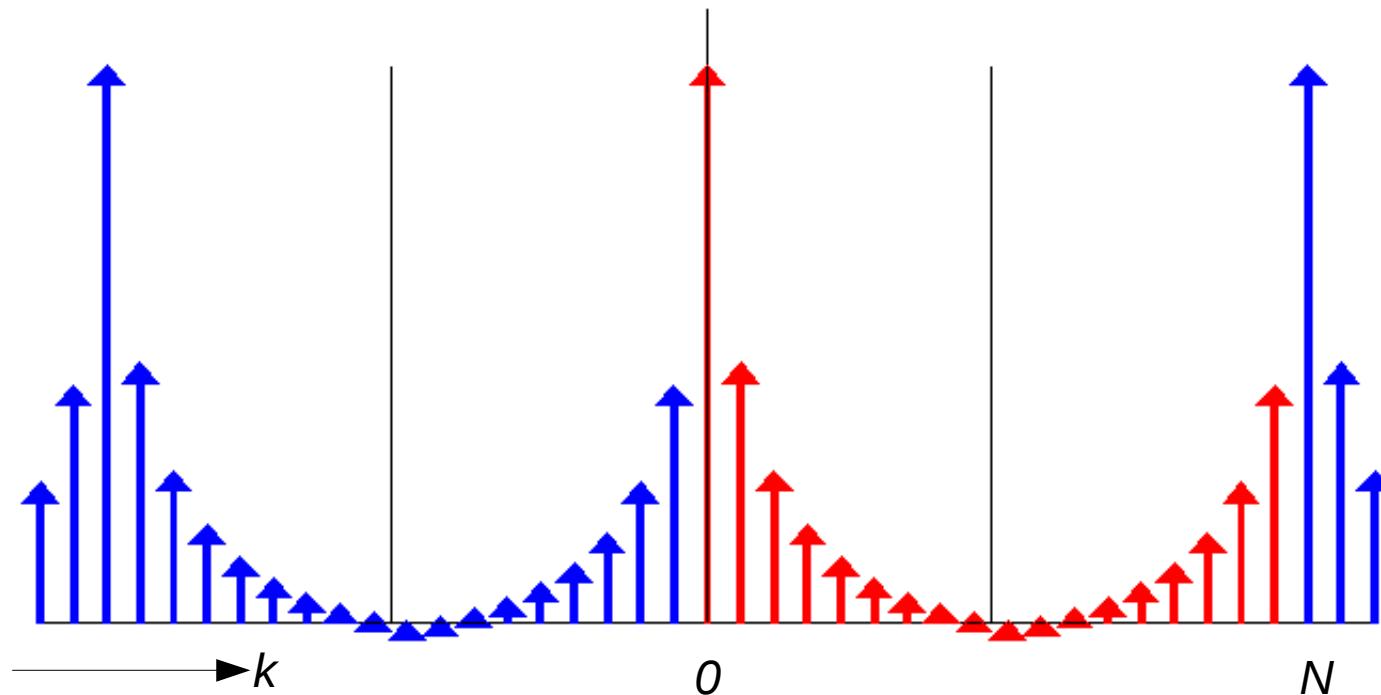
$$f[n] = \frac{1}{N} \sum_{k=0}^{N-1} F[k] e^{i \frac{2\pi}{N} kn}$$

Main difference with  $F(\omega)$  is that  $F[k]$  is defined on a limited domain ( $N$  samples), and that these samples are assumed to repeat periodically:  $F[k] = F[k+N]$ .

In the same way,  $f[n]$  is defined by  $N$  samples, which are assumed to repeat periodically:  $f[n] = f[n+N]$ .

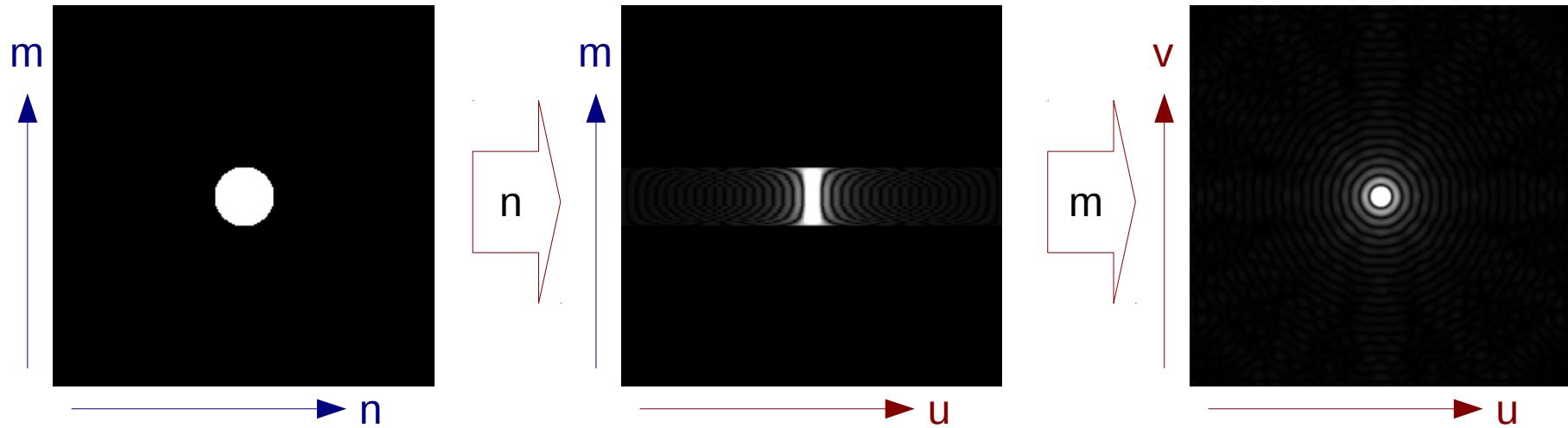
# Discrete Fourier transform

- The DFT only has positive frequencies !?!?!
- Remember: it is periodic!  $F[k] = F[k+N]$
- Thus:  $F[-k] = F[N-k]$



# Fourier transform in 2D, 3D, etc.

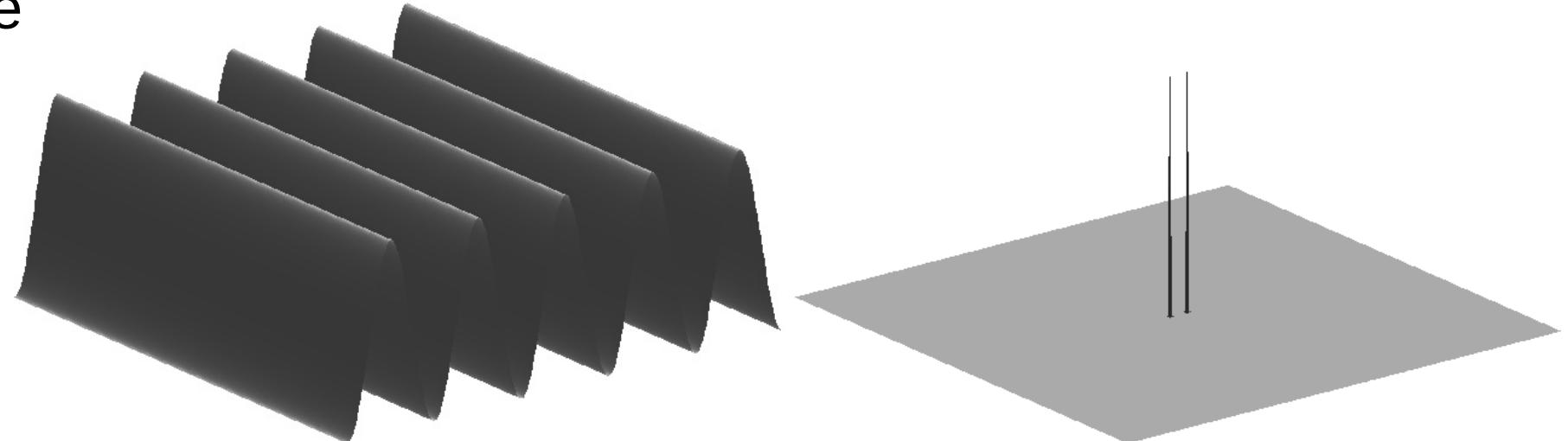
- Simplest thing there is! — the FT is separable:
  - Perform transform along x-axis,
  - Perform transform along y-axis of result,
  - Perform transform along z-axis of result, (etc.)



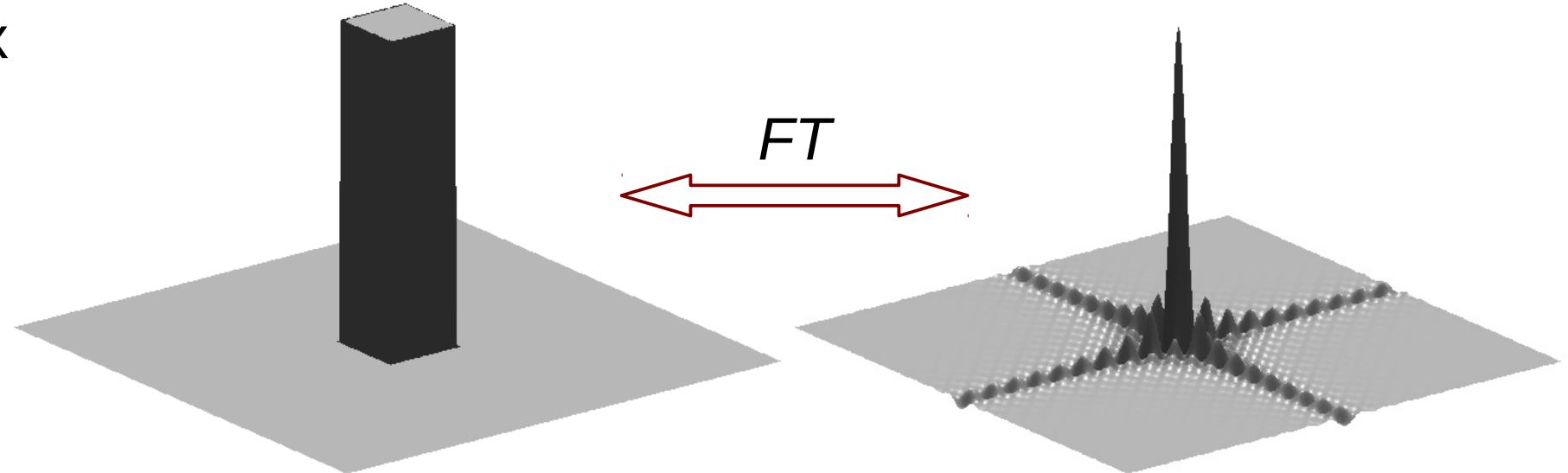
$$F[u, v] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[n, m] e^{-i 2\pi \left( \frac{un}{N} + \frac{vm}{M} \right)} = \sum_{m=0}^{M-1} \left( \sum_{n=0}^{N-1} f[n, m] e^{-i \frac{2\pi}{N} un} \right) e^{-i \frac{2\pi}{M} vm}$$

# 2D Fourier transform pairs

sine

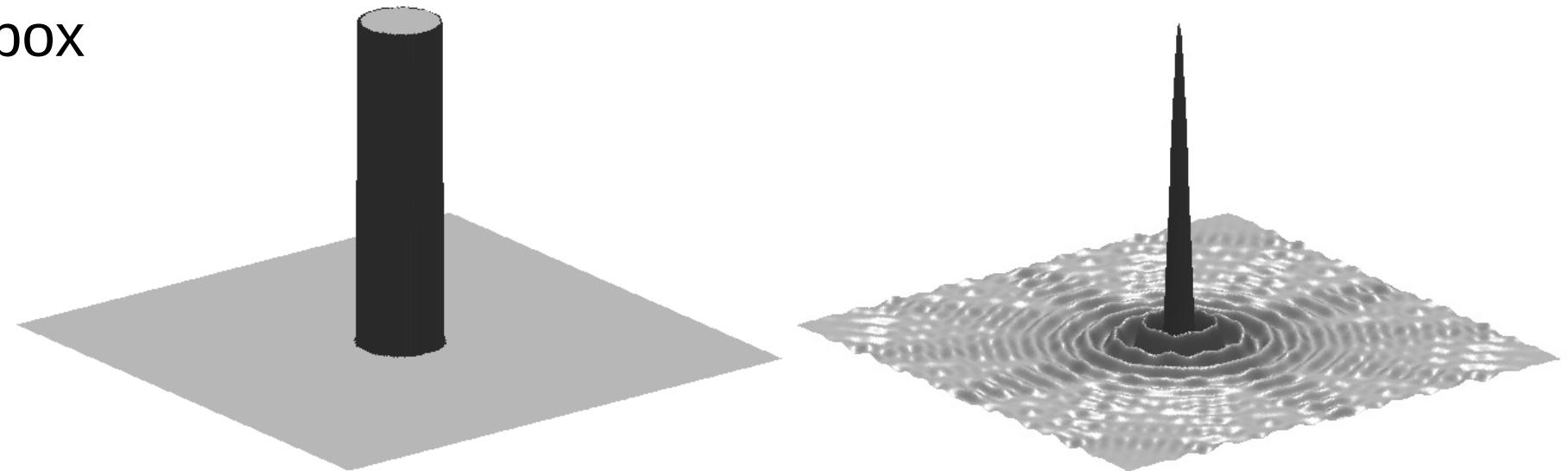


box

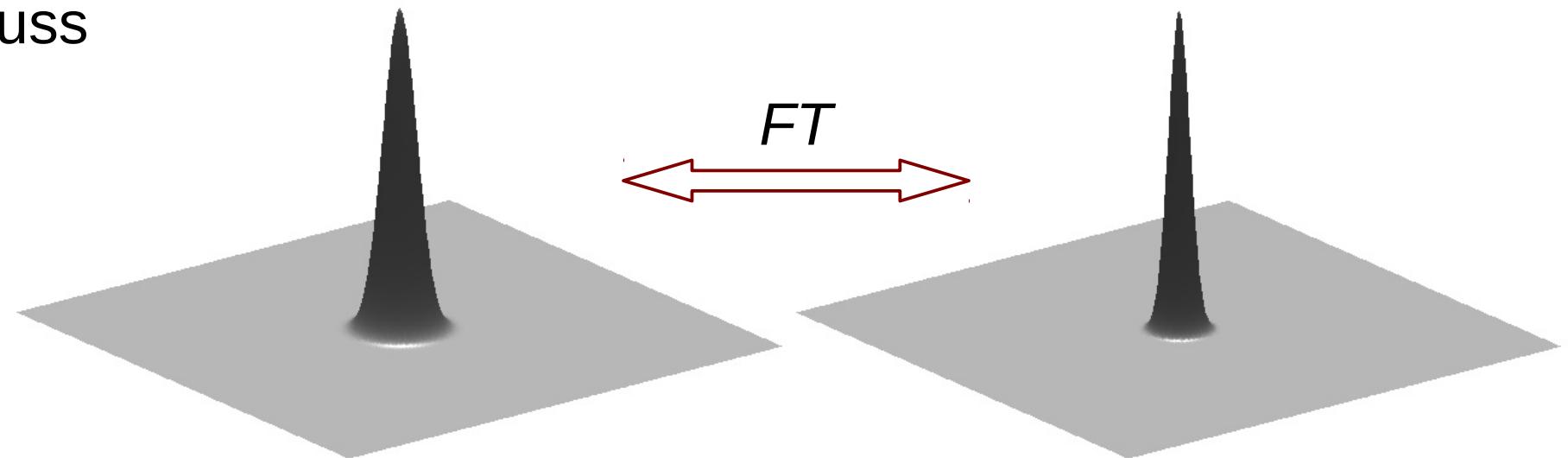


# 2D Fourier transform pairs

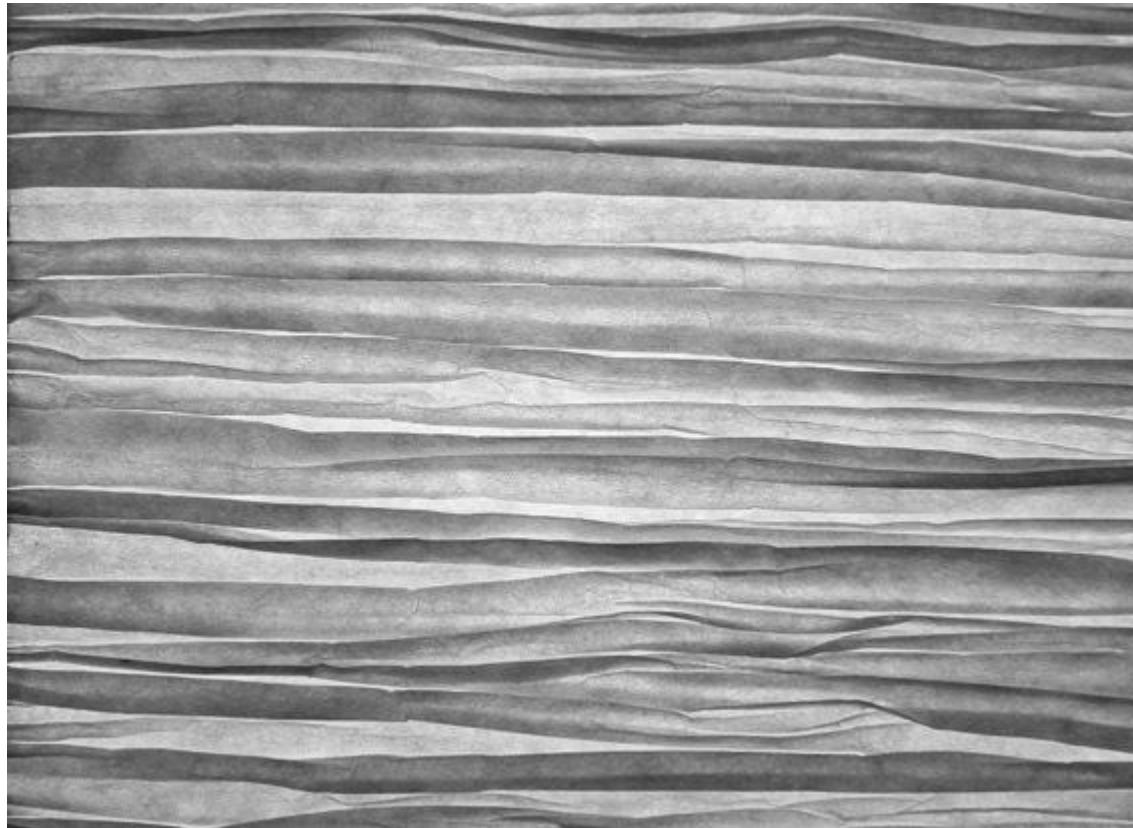
pillbox



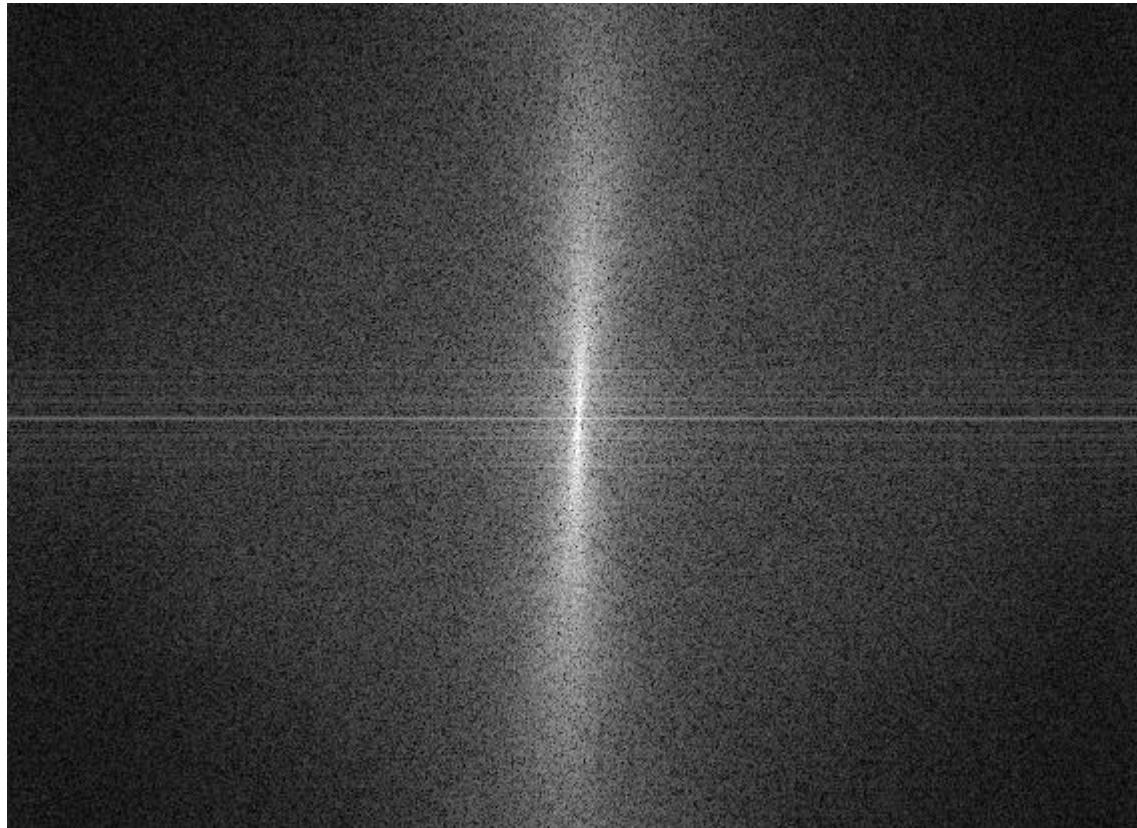
Gauss



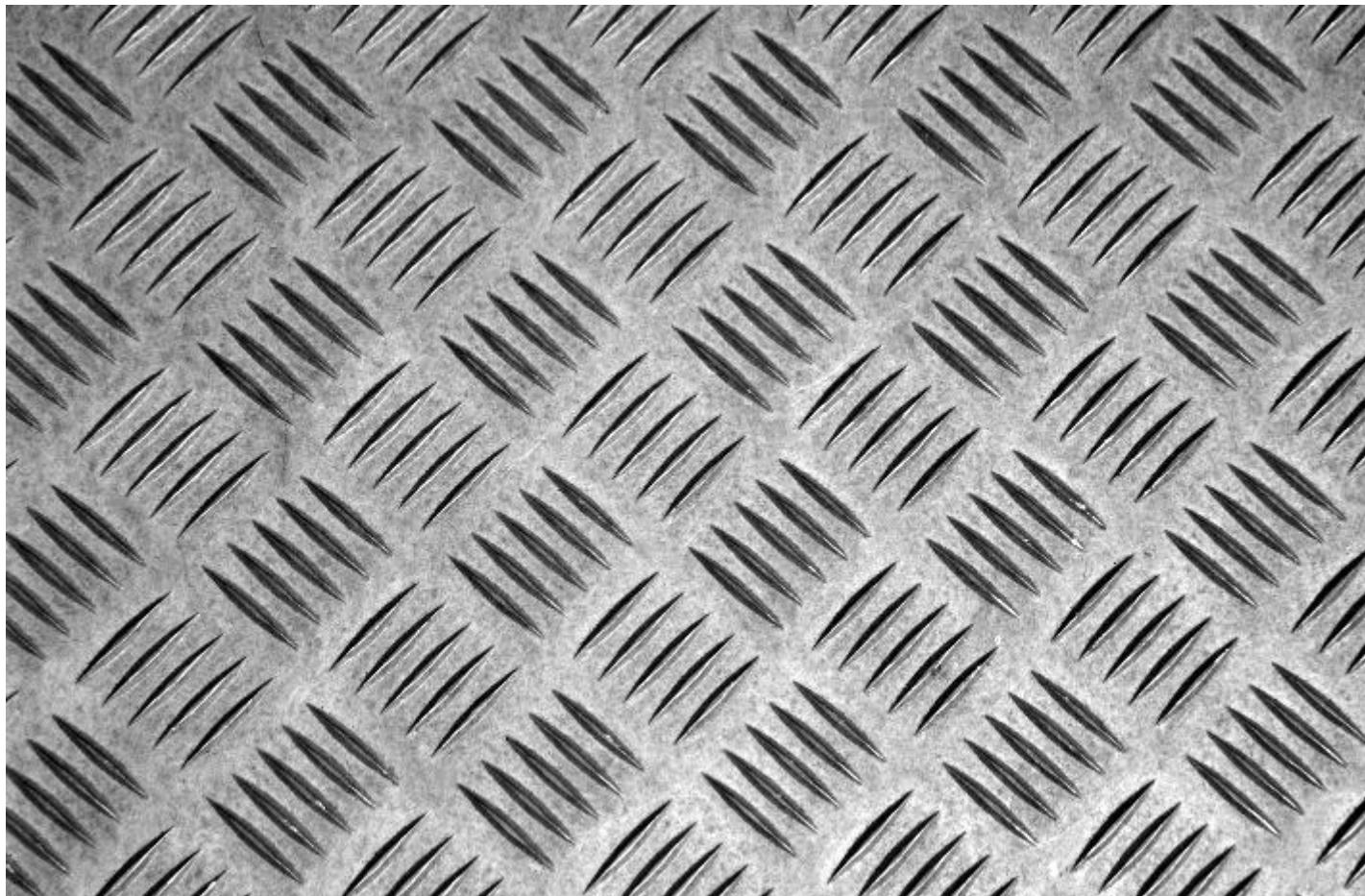
# 2D transform example 1



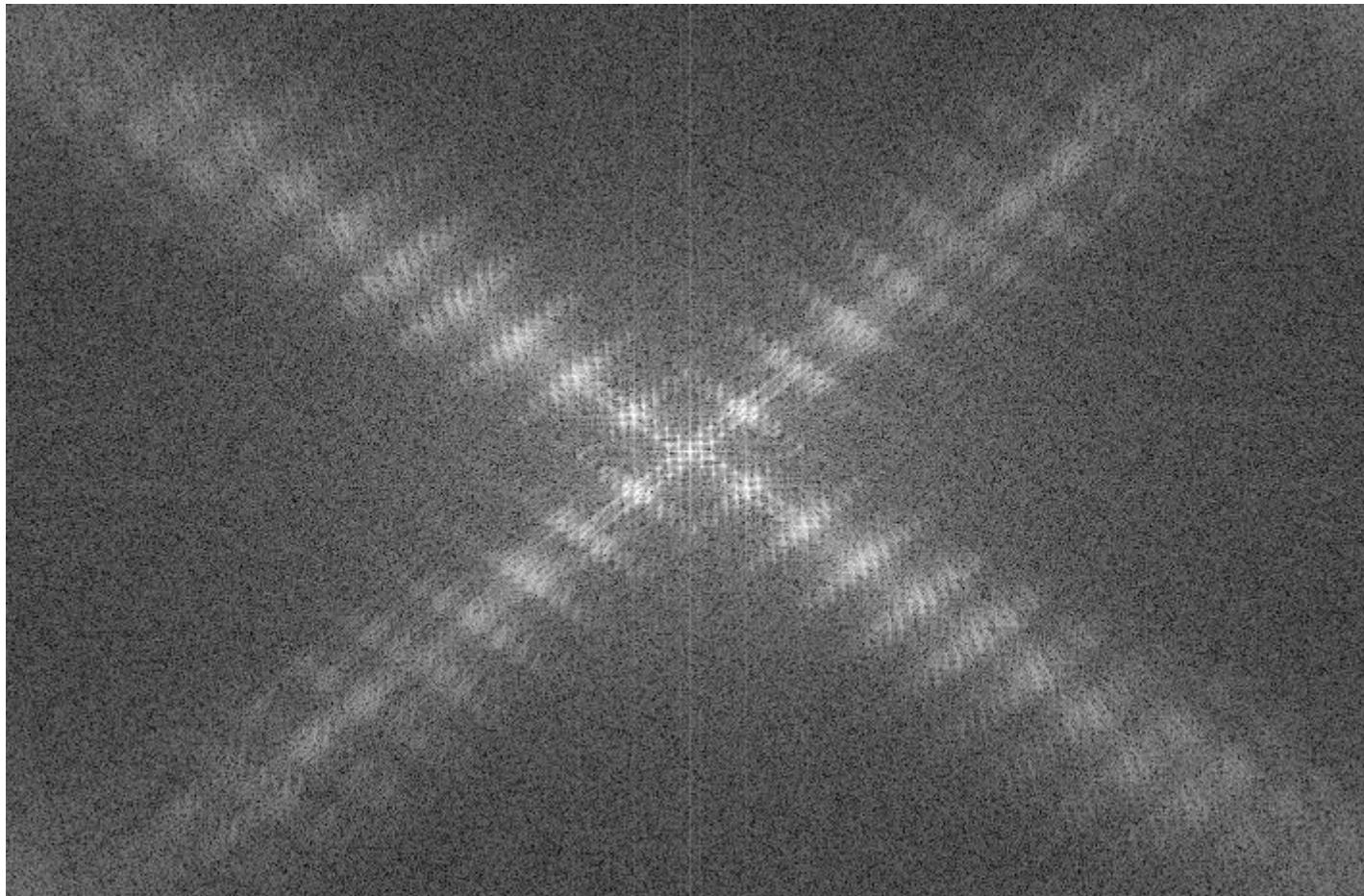
# 2D transform example 1



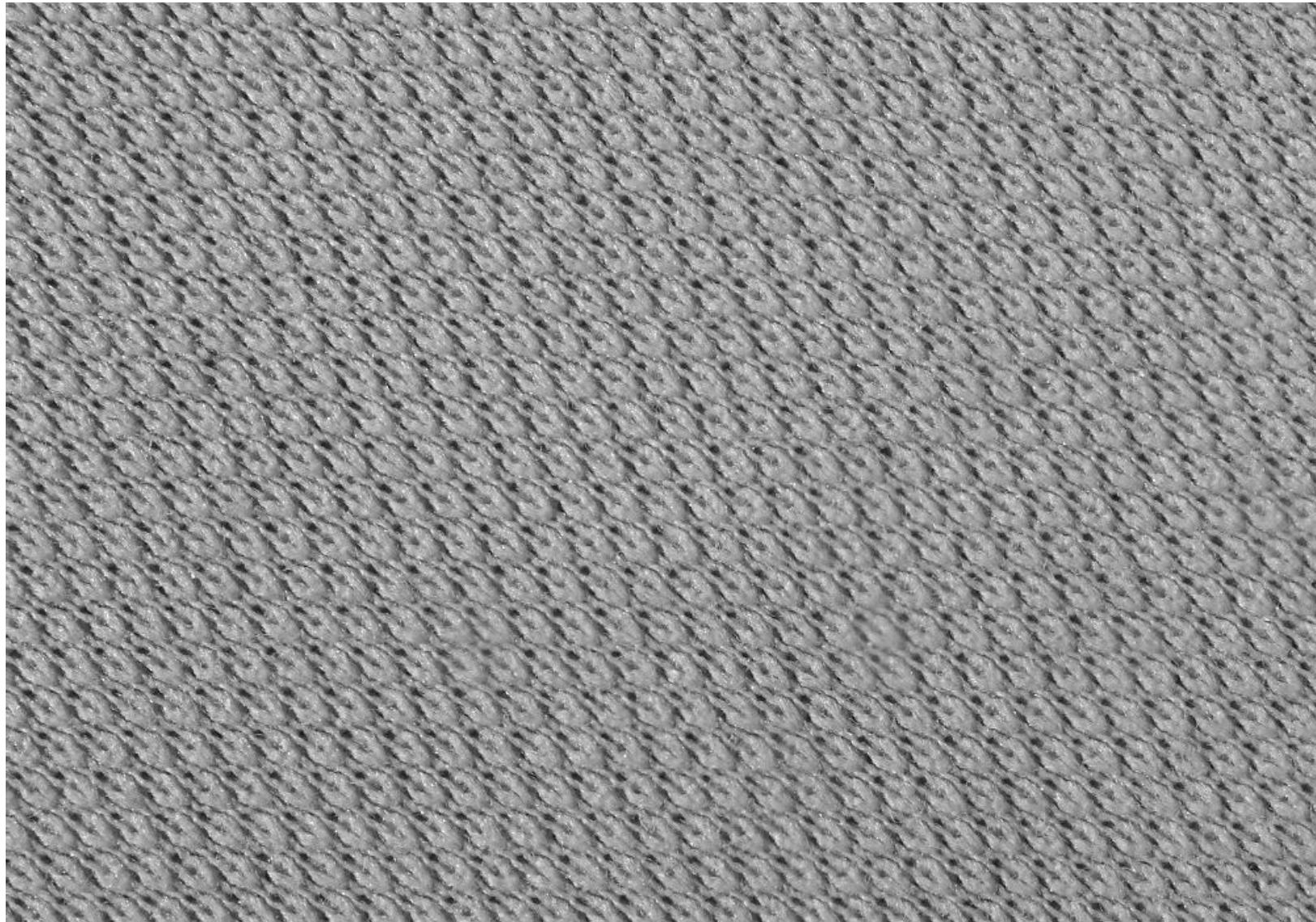
# 2D transform example 2



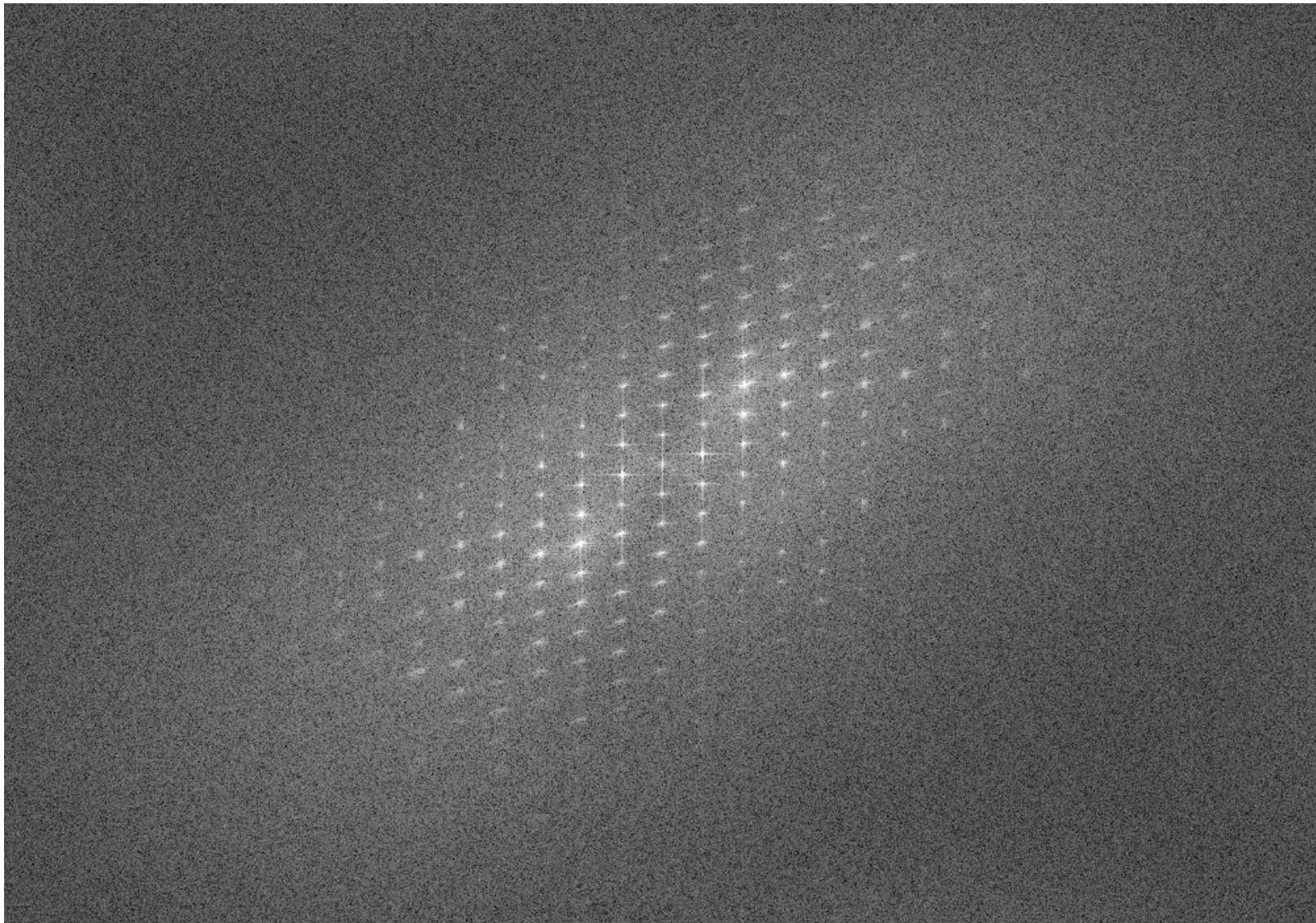
# 2D transform example 2



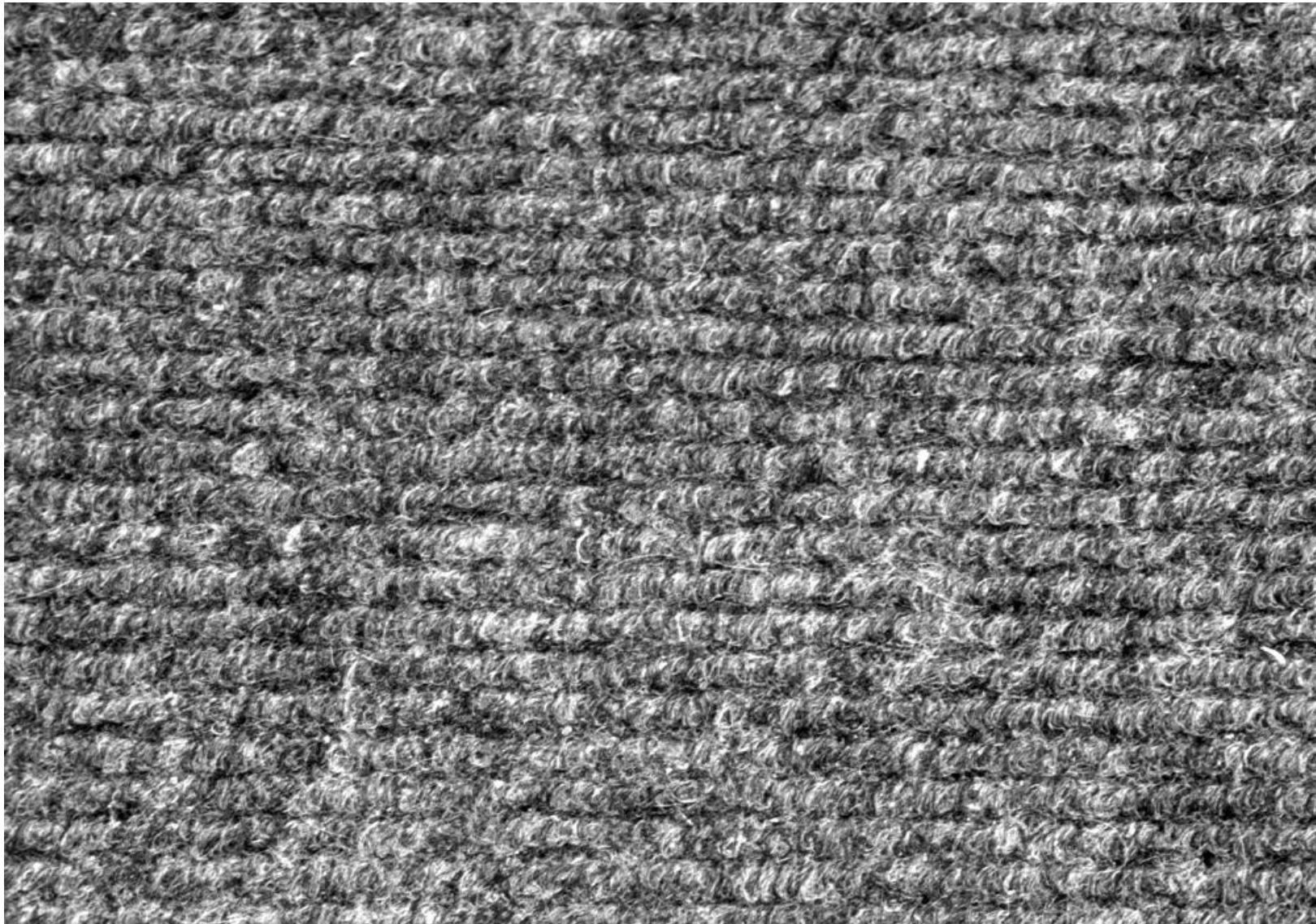
# 2D transform example 3



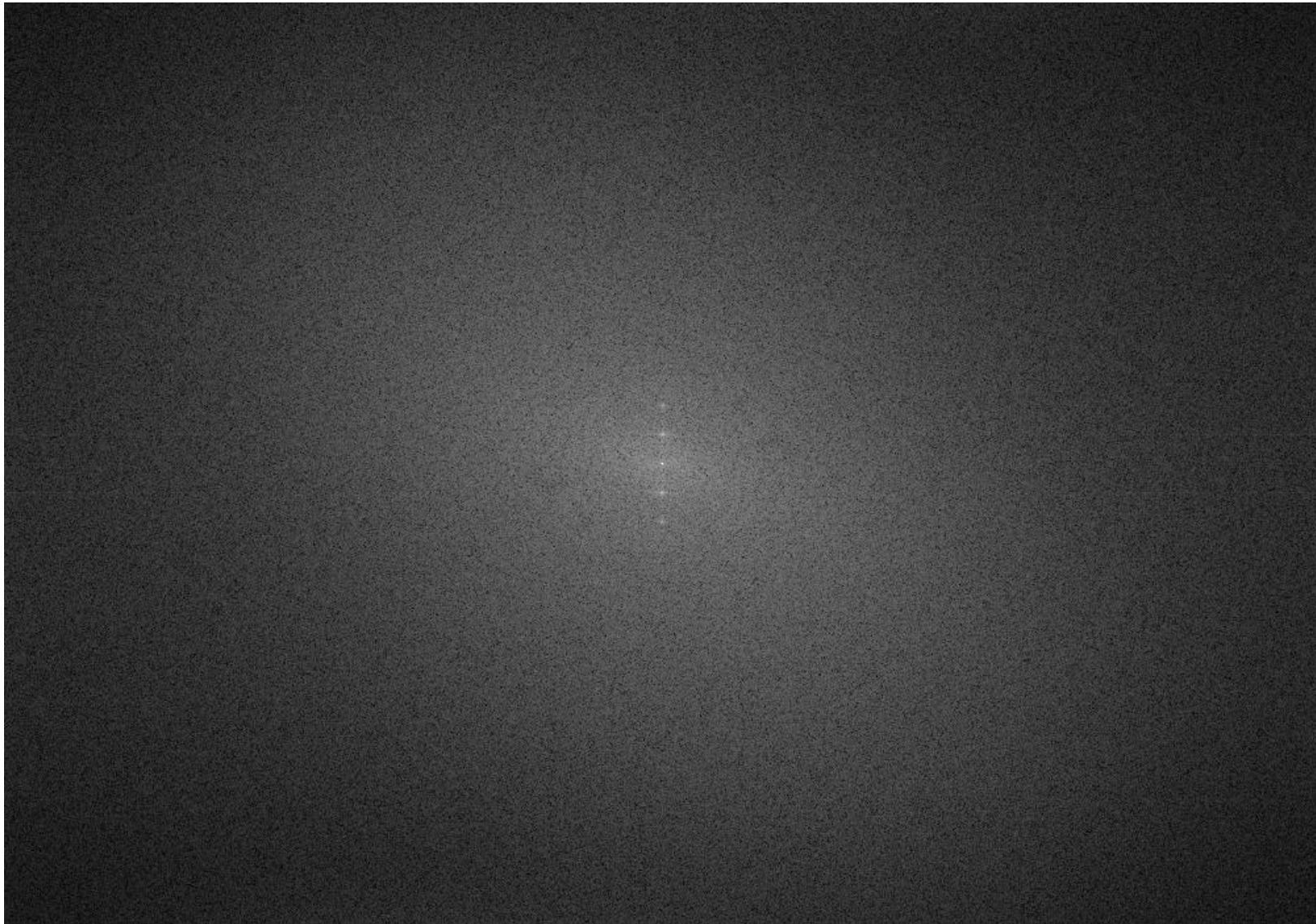
# 2D transform example 3



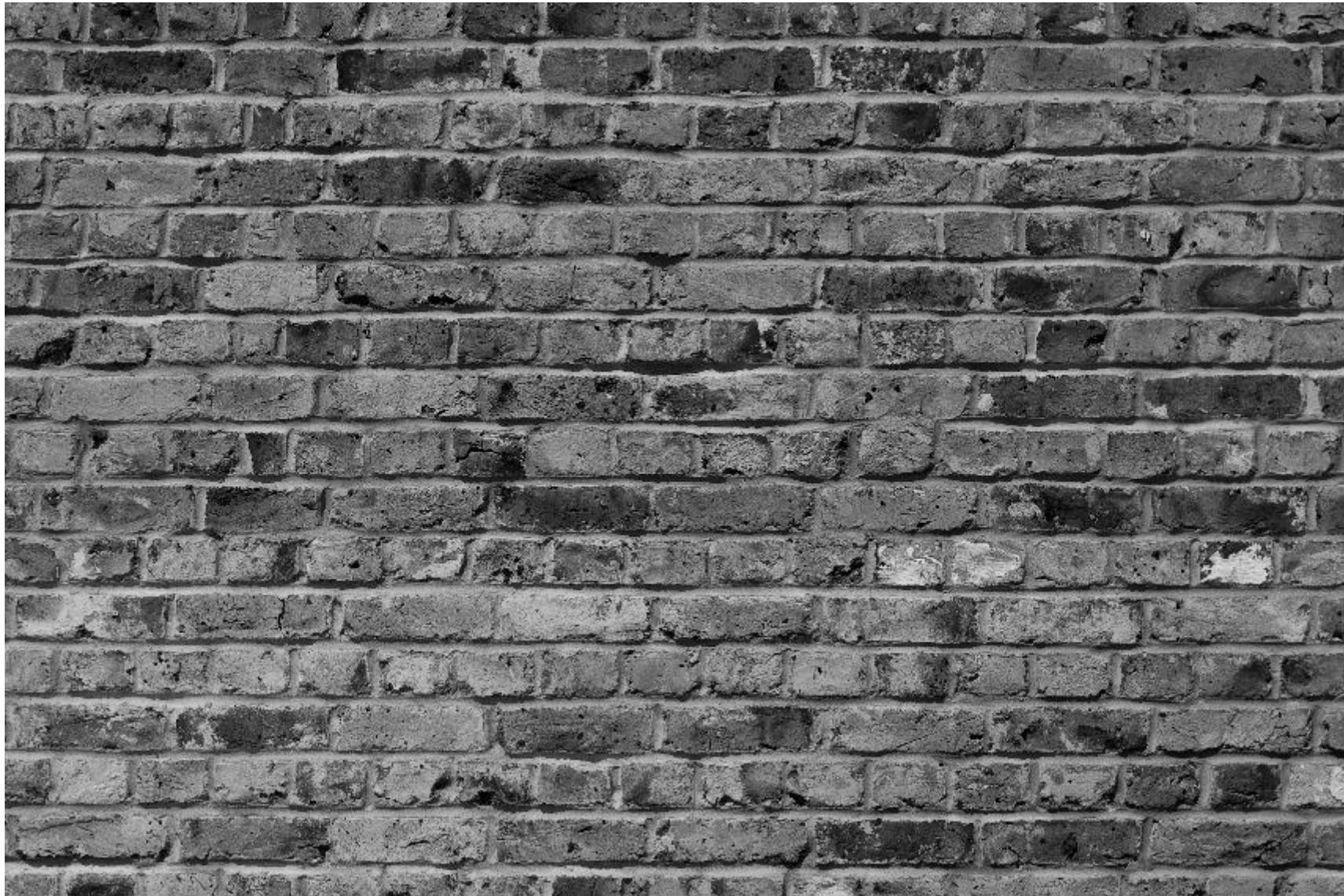
# 2D transform example 4



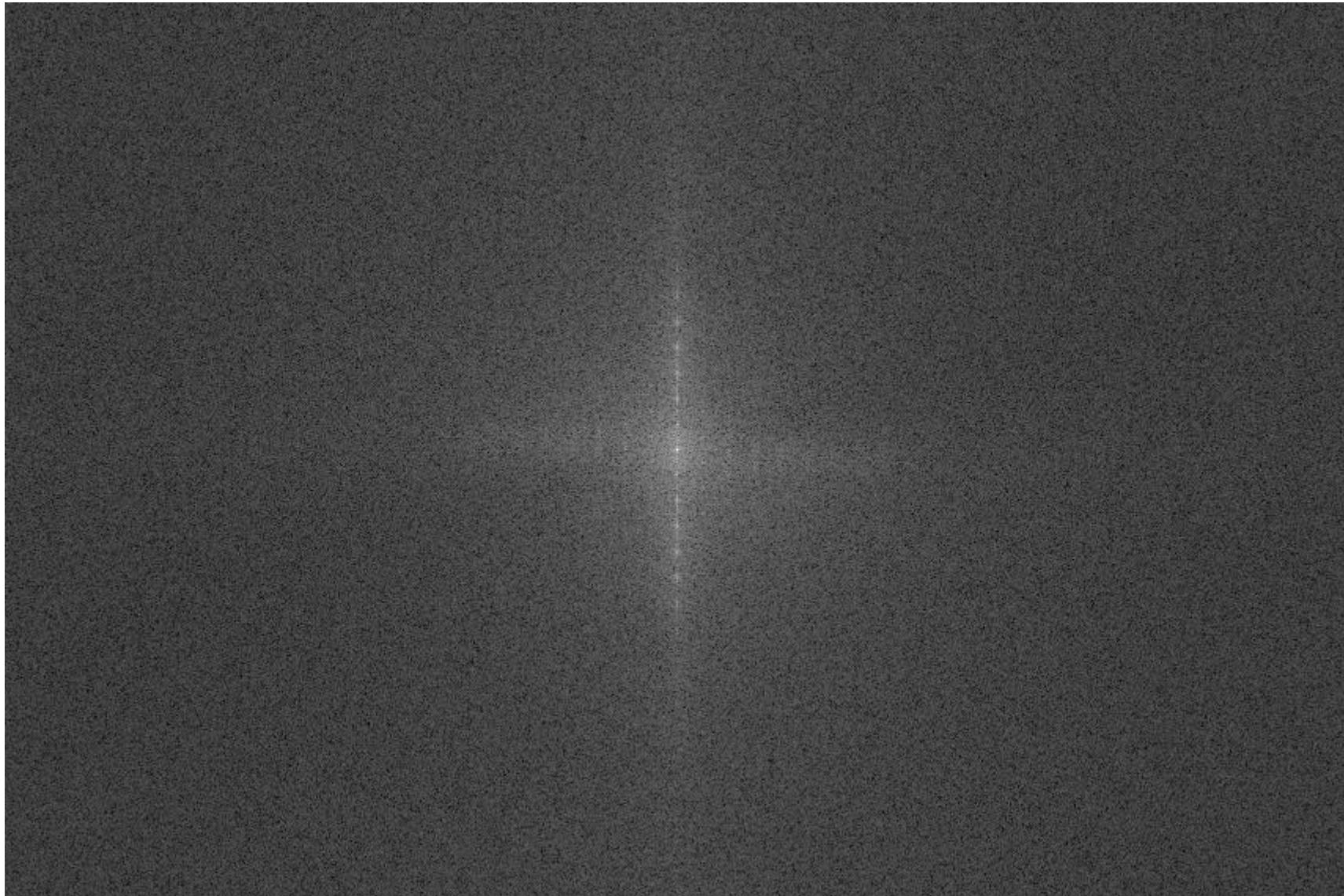
# 2D transform example 4



# 2D transform example 5



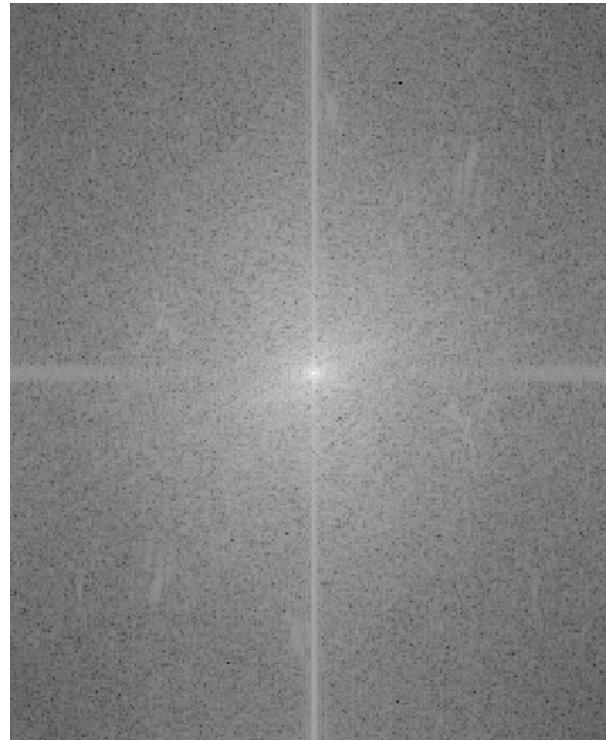
# 2D transform example 5



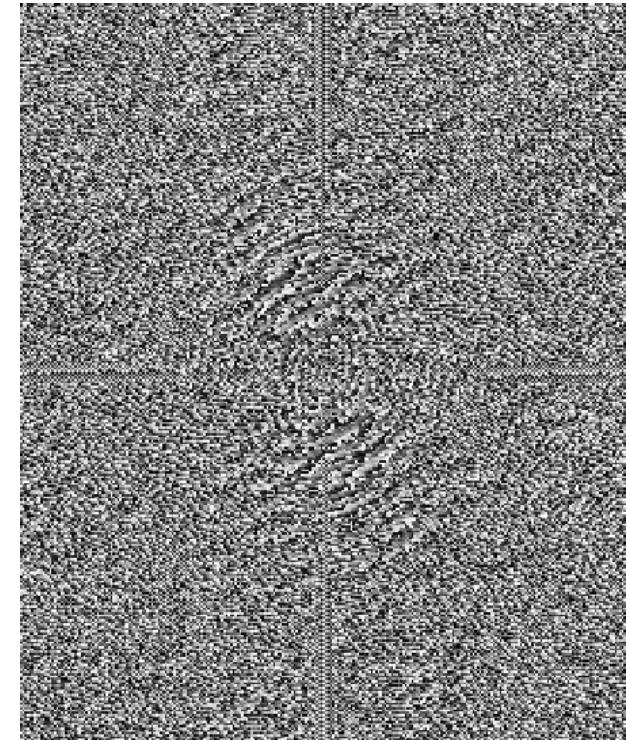
# What is more important?



Jean Baptiste Joseph Fourier

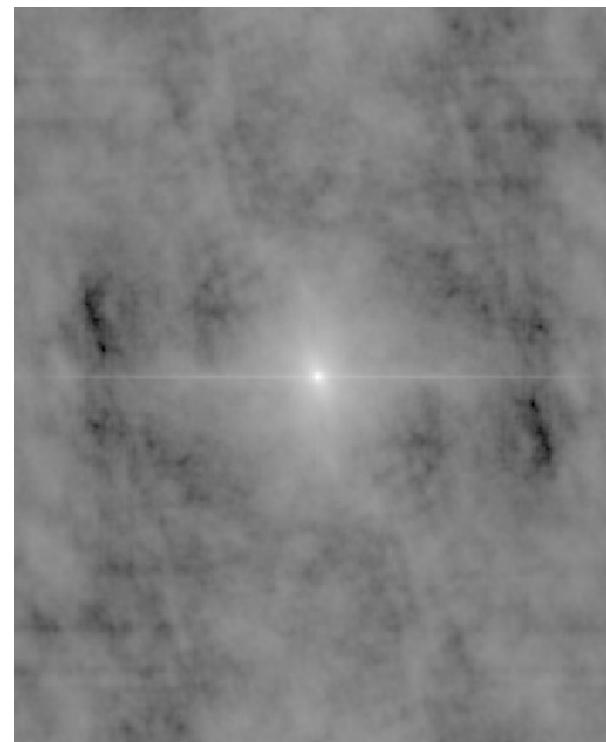
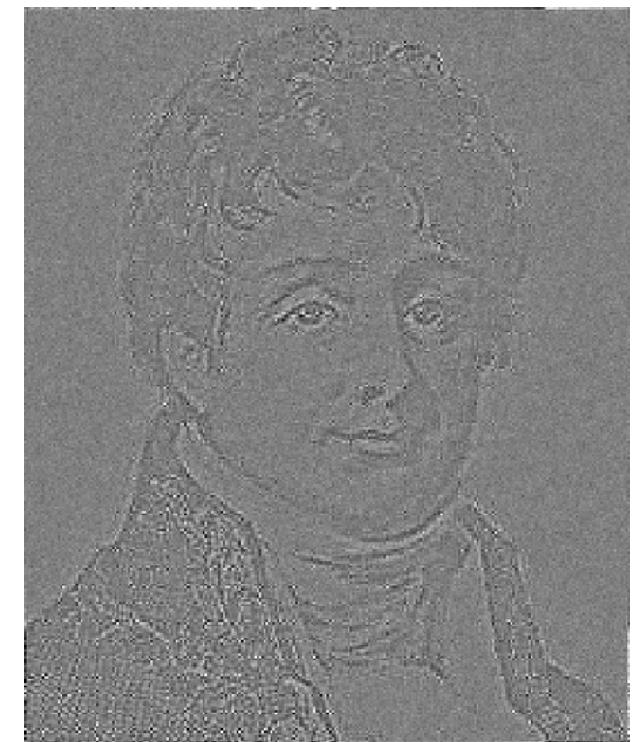
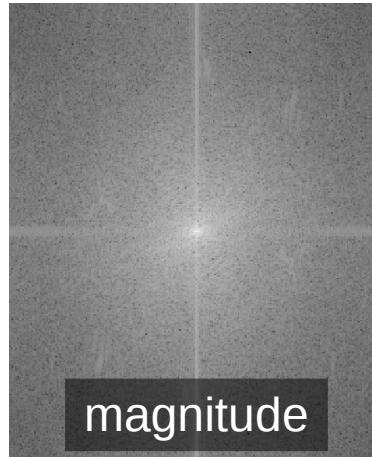


magnitude



phase

# What is more important?



# The Fast Fourier Transform (FFT)

- Clever algorithm to compute the DFT.
- Runs in  $O(N \log N)$  time, rather than  $O(N^2)$  time.
- Because of symmetry of the forward and inverse Fourier transforms, FFT can also compute the IDFT.

$$F[k] = F_{\text{even}}[k] + F_{\text{odd}}[k]e^{-i\frac{2\pi}{N}k} \quad N = 2M$$

$$F[k+M] = F_{\text{even}}[k] - F_{\text{odd}}[k]e^{-i\frac{2\pi}{N}k}$$



# Convolution in the Fourier domain

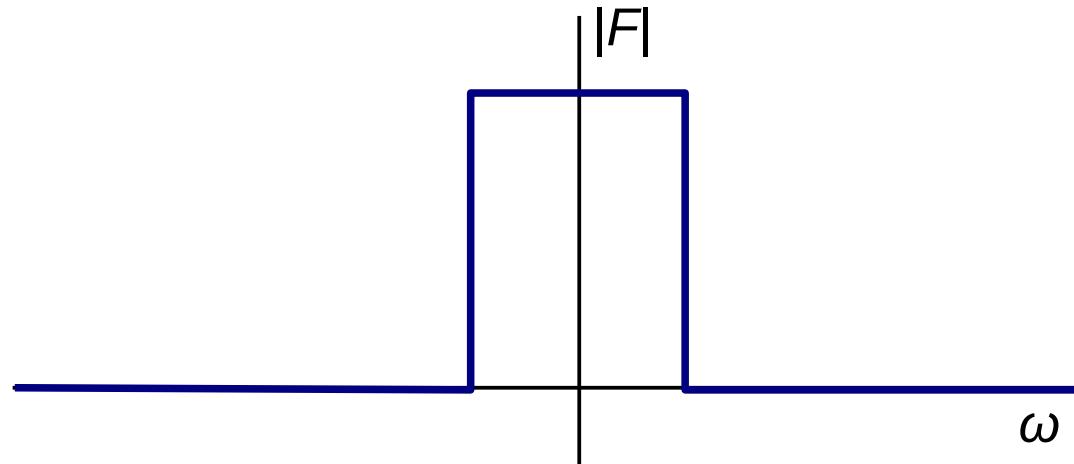
- The Convolution property of the Fourier transform:

$$\mathcal{F}\{f \otimes h\} = \mathcal{F}\{f\} \cdot \mathcal{F}\{h\}$$

- Thus we can calculate the convolution through:
  - $F = \text{FFT}(f)$
  - $H = \text{FFT}(h)$
  - $G = F \cdot H$
  - $g = \text{IFFT}(G)$
- Convolution is an operation of  $O(NM)$ 
  - $N$  image pixels,  $M$  kernel pixels
- Through the FFT it is an operation of  $O(N \log N)$ 
  - Efficient if  $M$  is large!

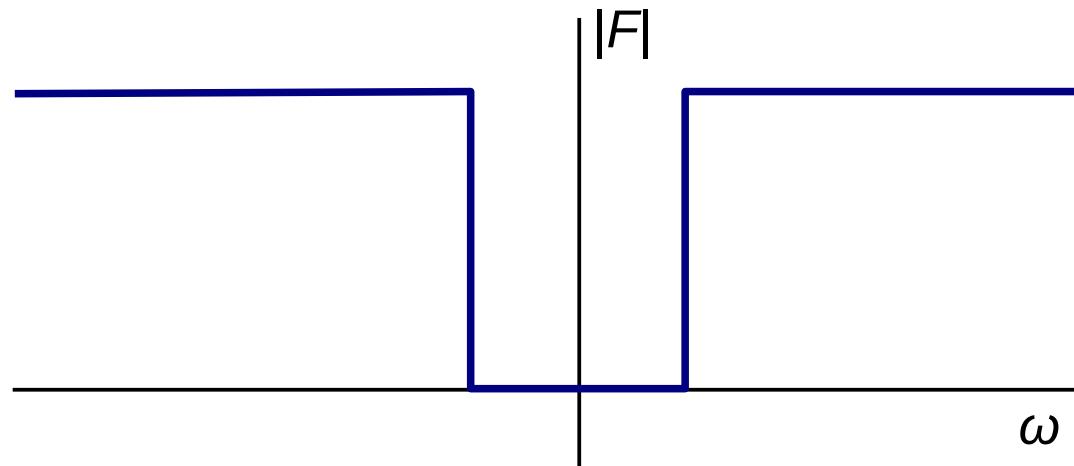
# Low-pass filtering

- Linear smoothing filters are all low-pass filters.
  - Mean filter (uniform weights)
  - Gauss filter (Gaussian weights)
- Low-pass means low frequencies are not altered, high frequencies are attenuated



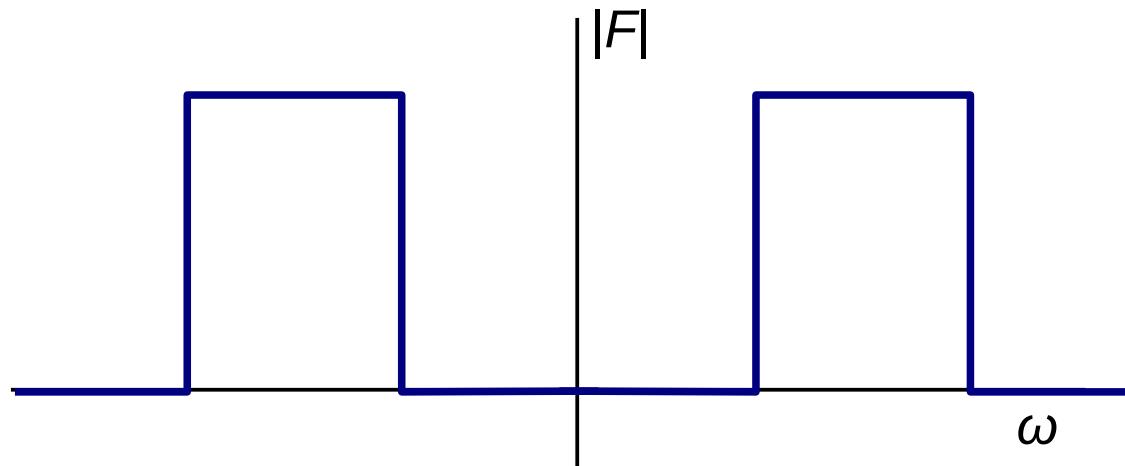
# High-pass filtering

- The opposite of low-pass filtering: low frequencies are attenuated, high frequencies are not altered
- The “unsharp mask” filter is a high-pass filter
- The Laplace filter is a high-pass filter



# Band-pass filtering

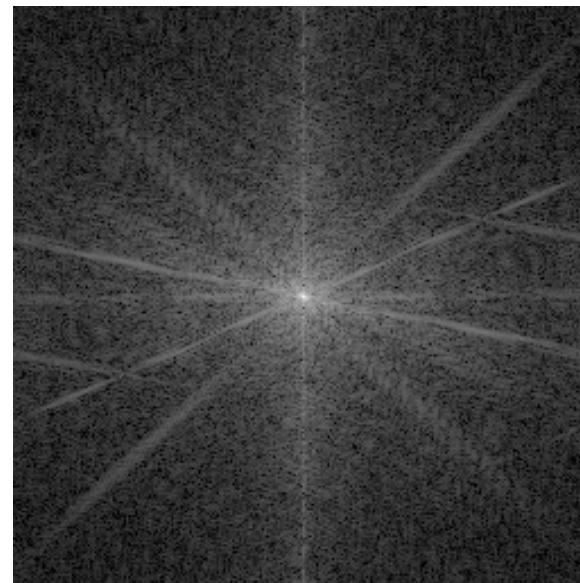
- You can choose any part of the frequency axis to preserve (band-pass filter).
- Or you can attenuate a specific set of frequencies (band-stop filter).



# Example: low-pass filtering

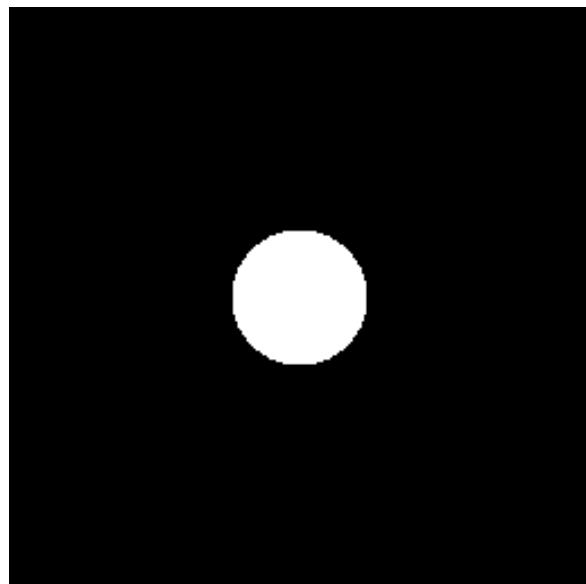


input image  $f$

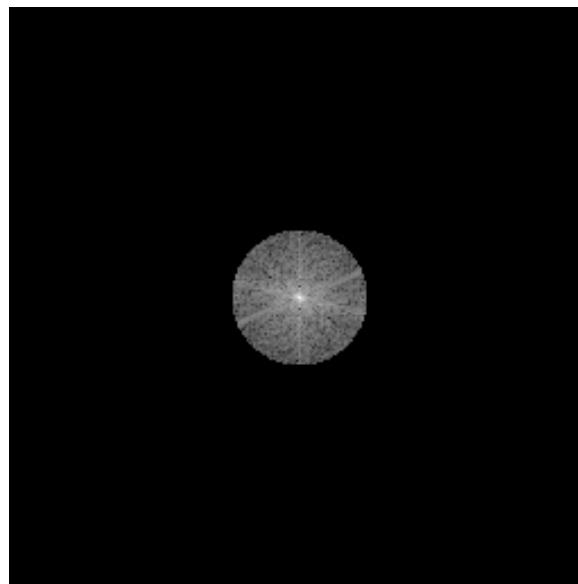


Fourier transform  $F$

# Example: frequency domain filtering



Fourier filter  $H$

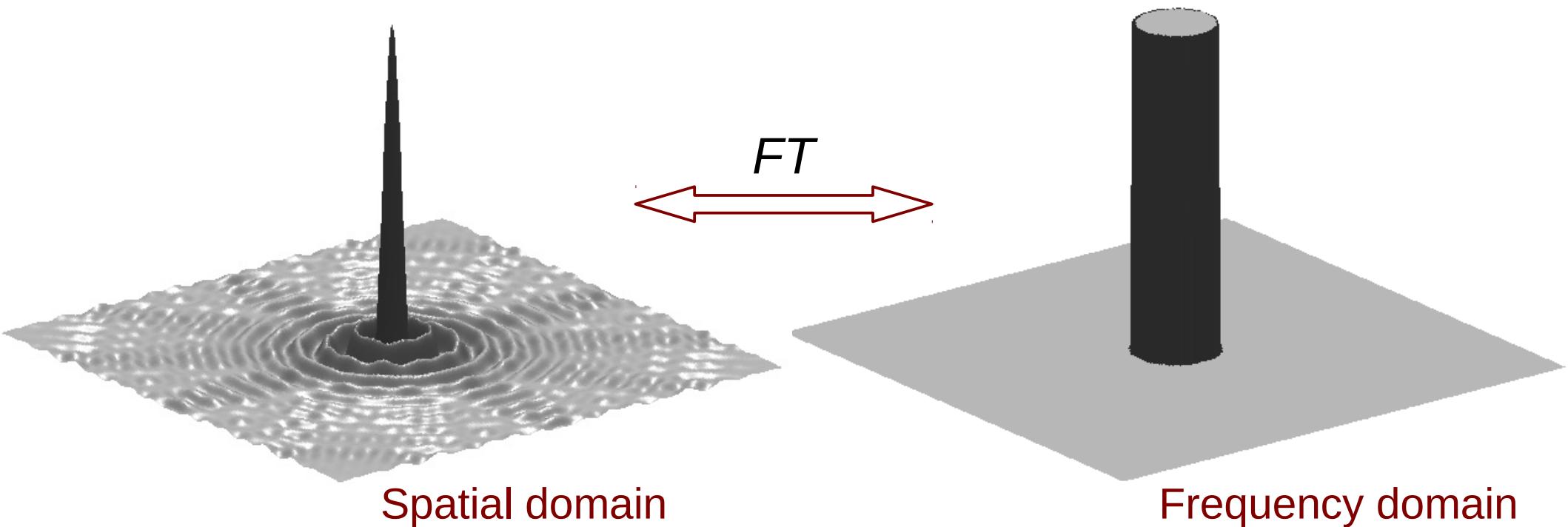


$G = F \cdot H$

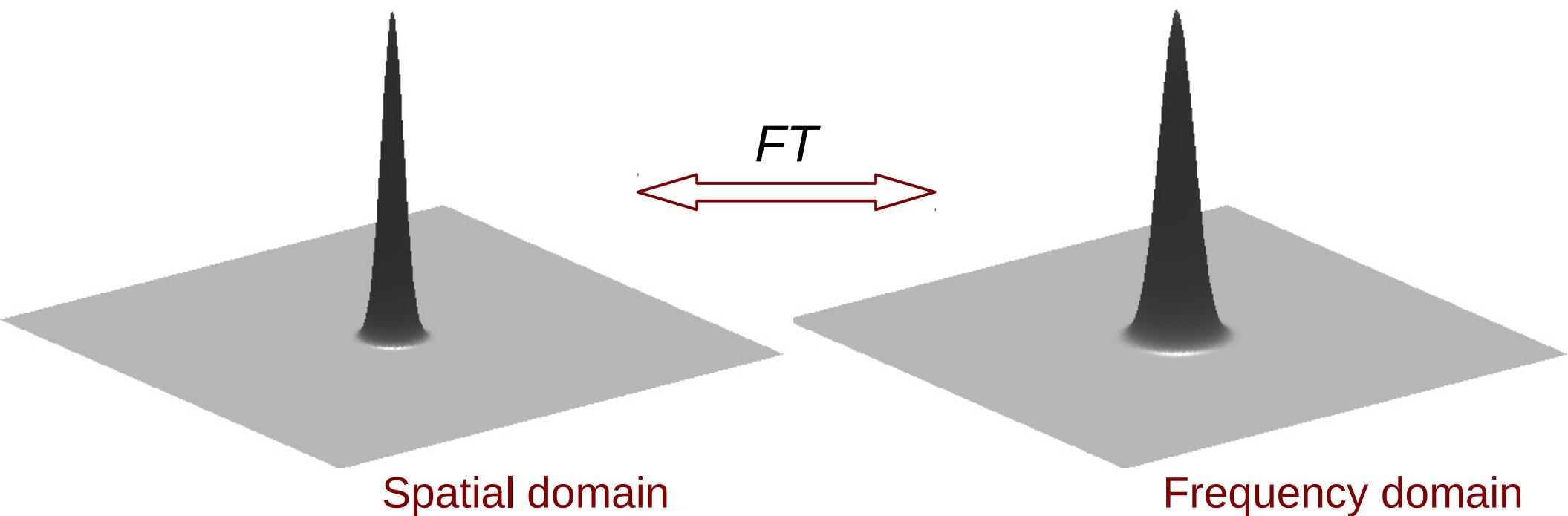


filtered image  $g$

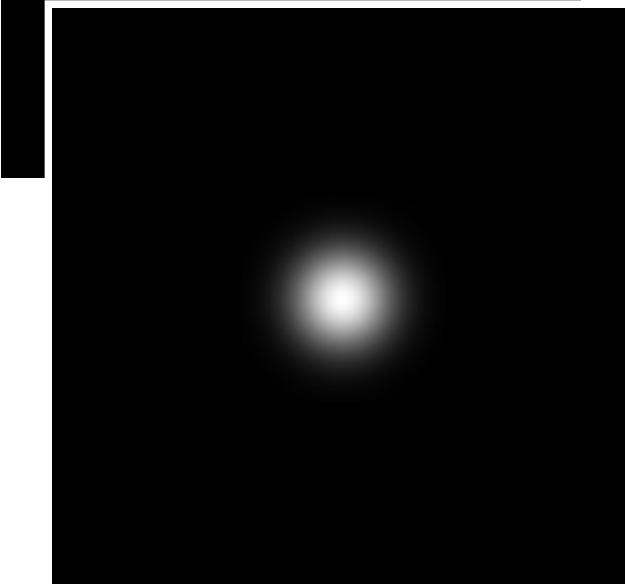
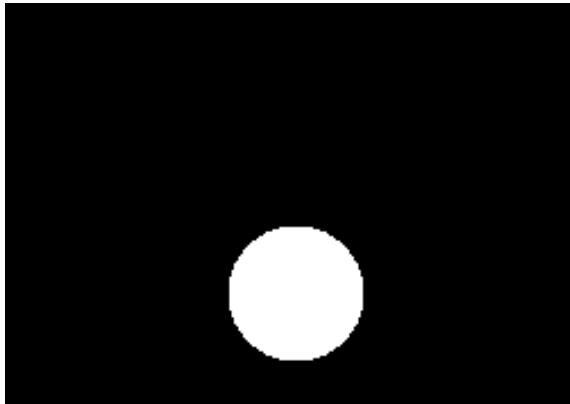
# Why the ringing?



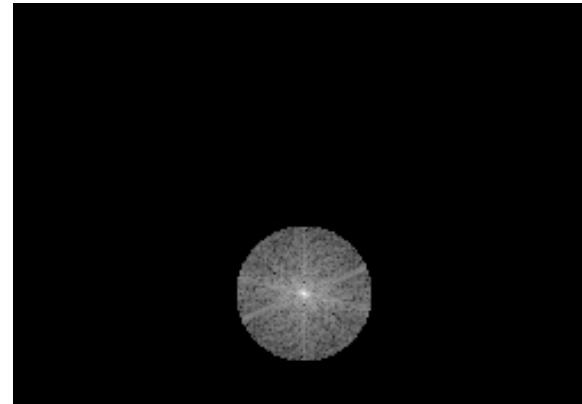
# What is the solution?



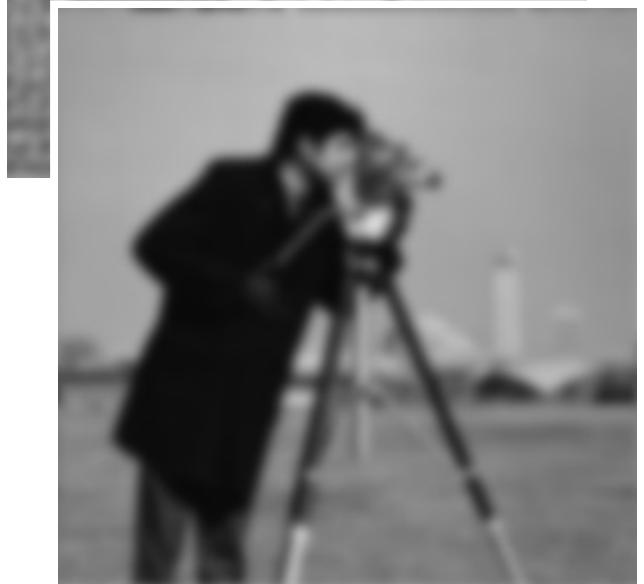
# Example: frequency domain filtering



Fourier filter  $H$

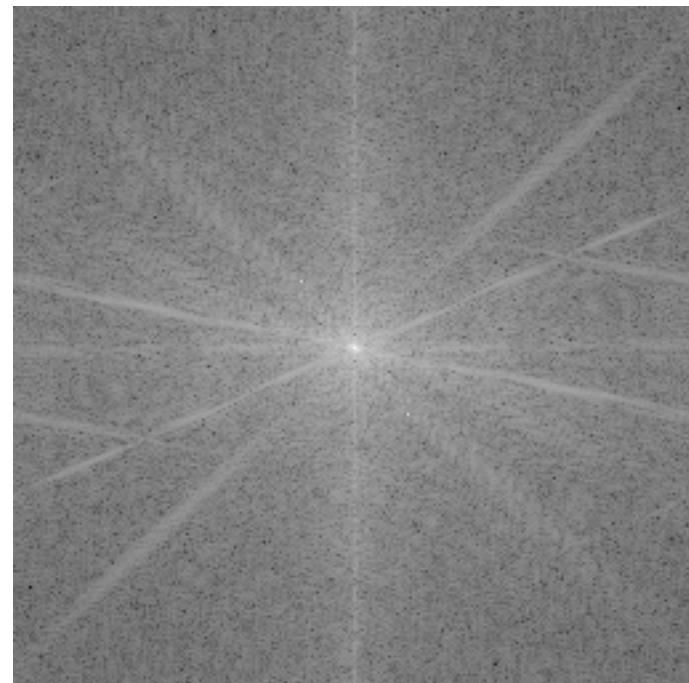


$G = F \cdot H$

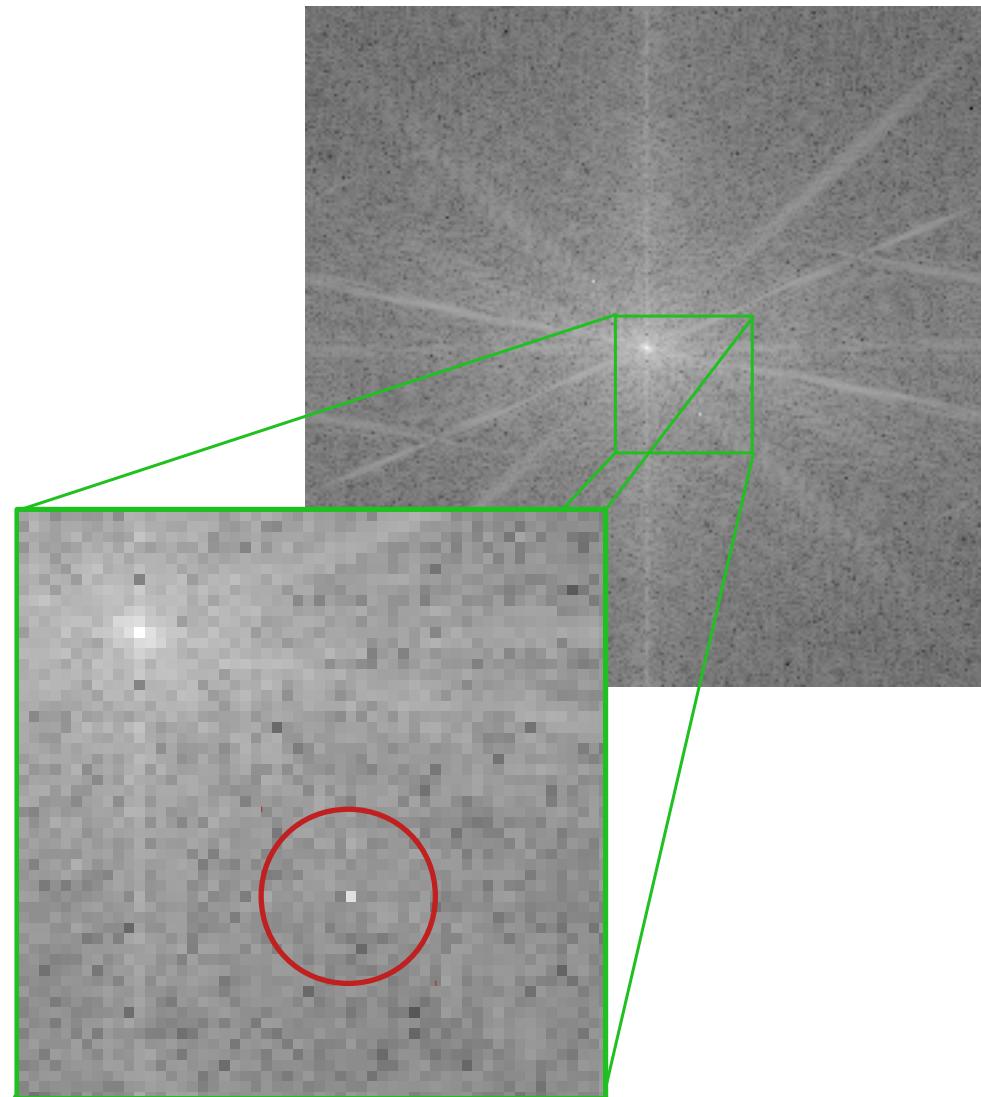


filtered image  $g$

# Structured noise

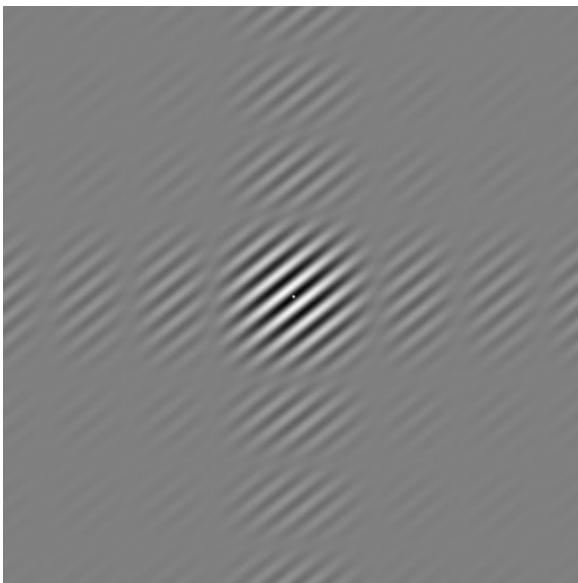
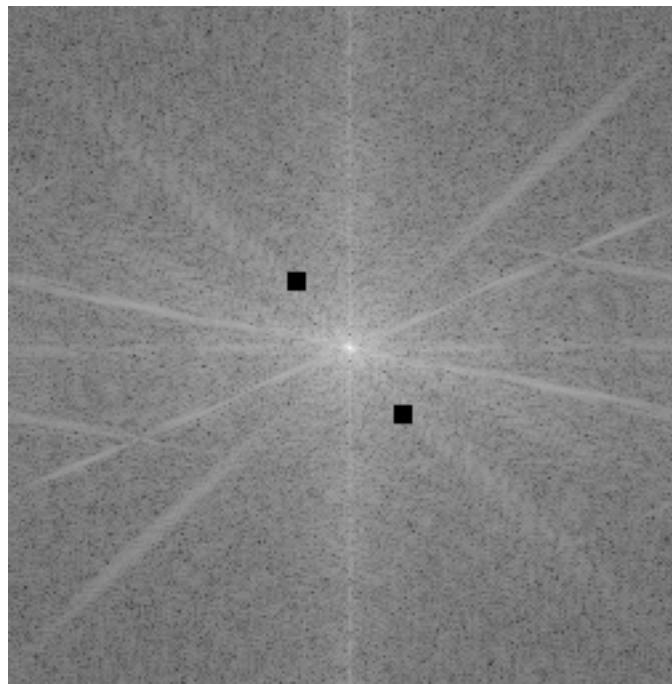
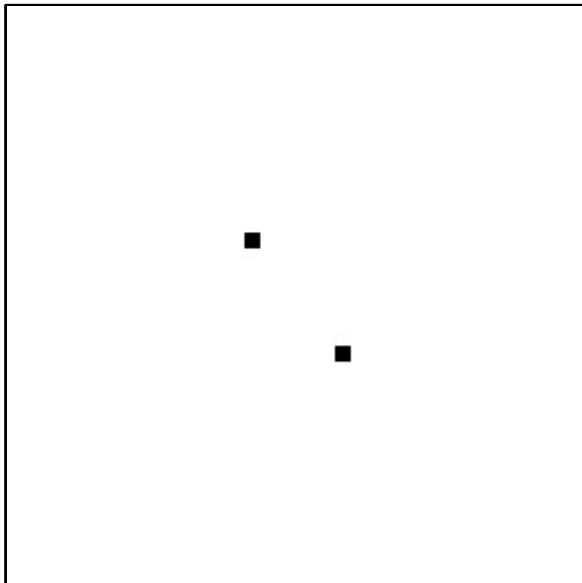


# Structured noise



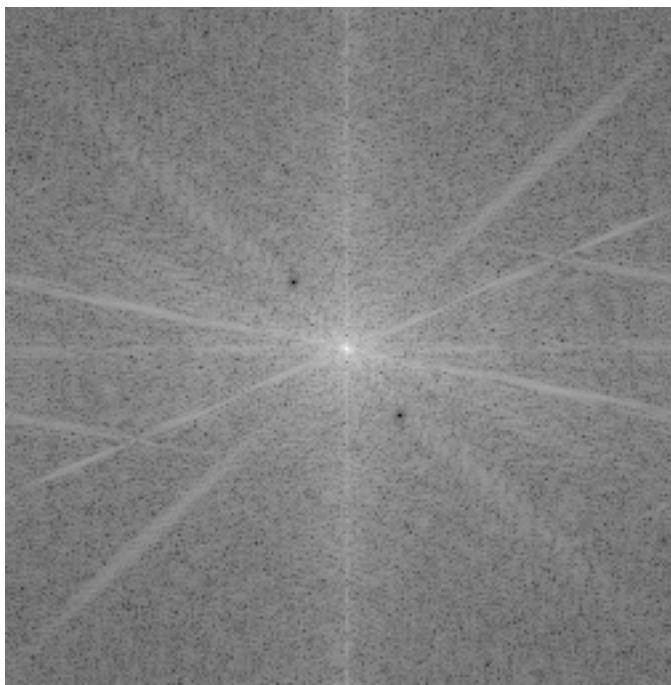
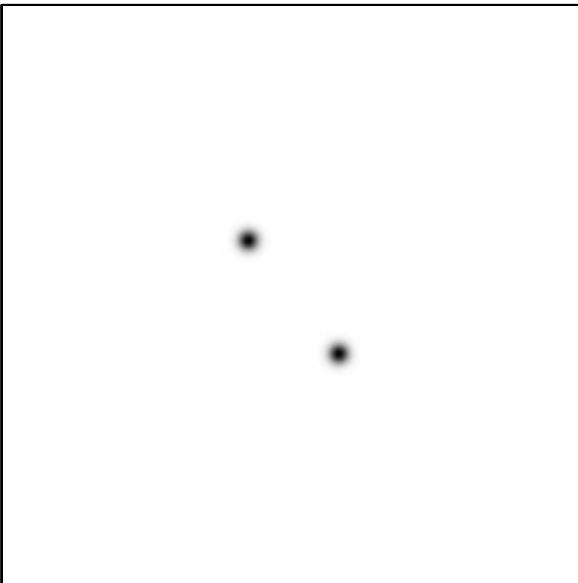
# Filtering structured noise

Notch filter



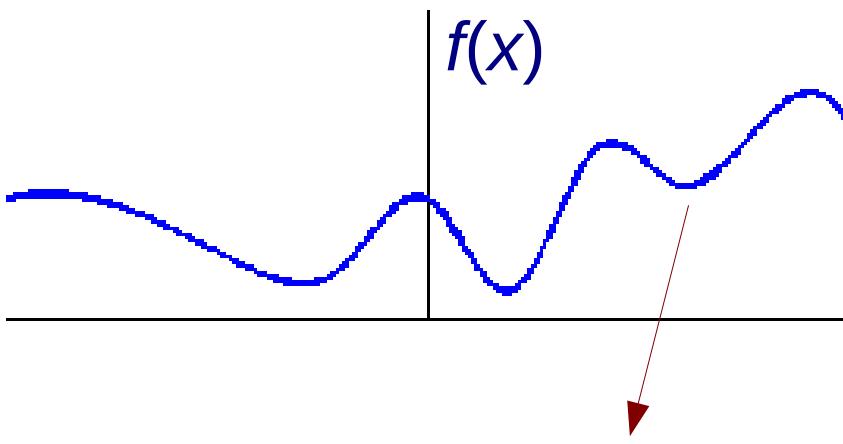
# Filtering structured noise

Notch filter, Gaussian

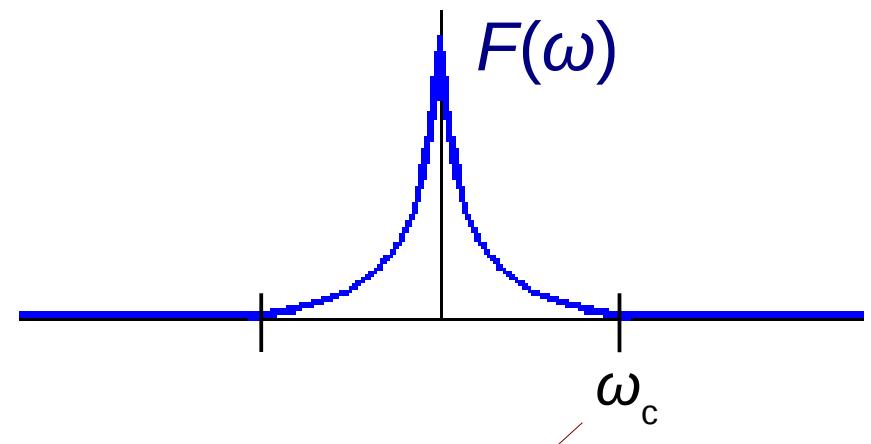


# Fourier analysis of sampling

$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-i\omega x} dx$$

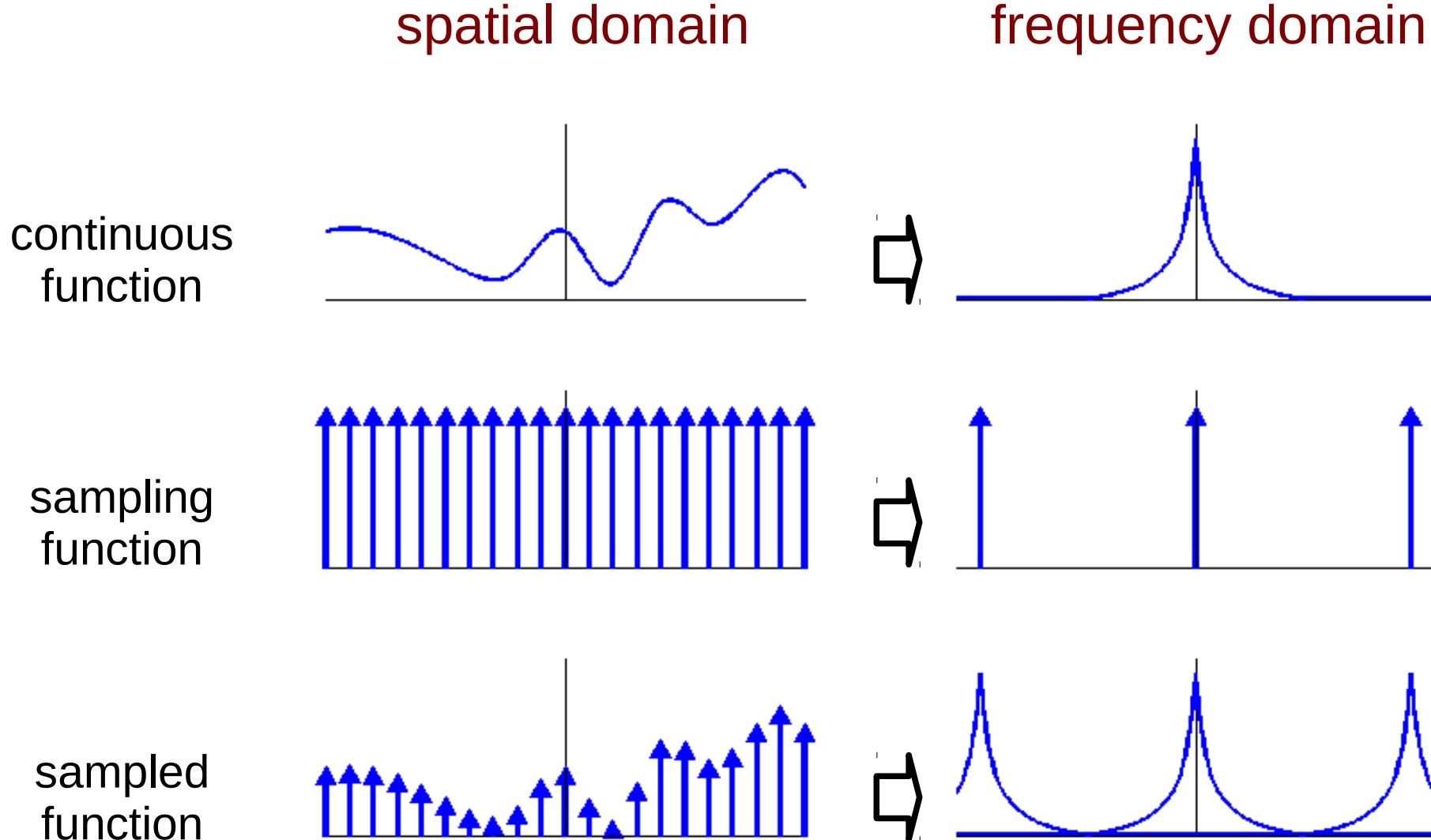


smooth function

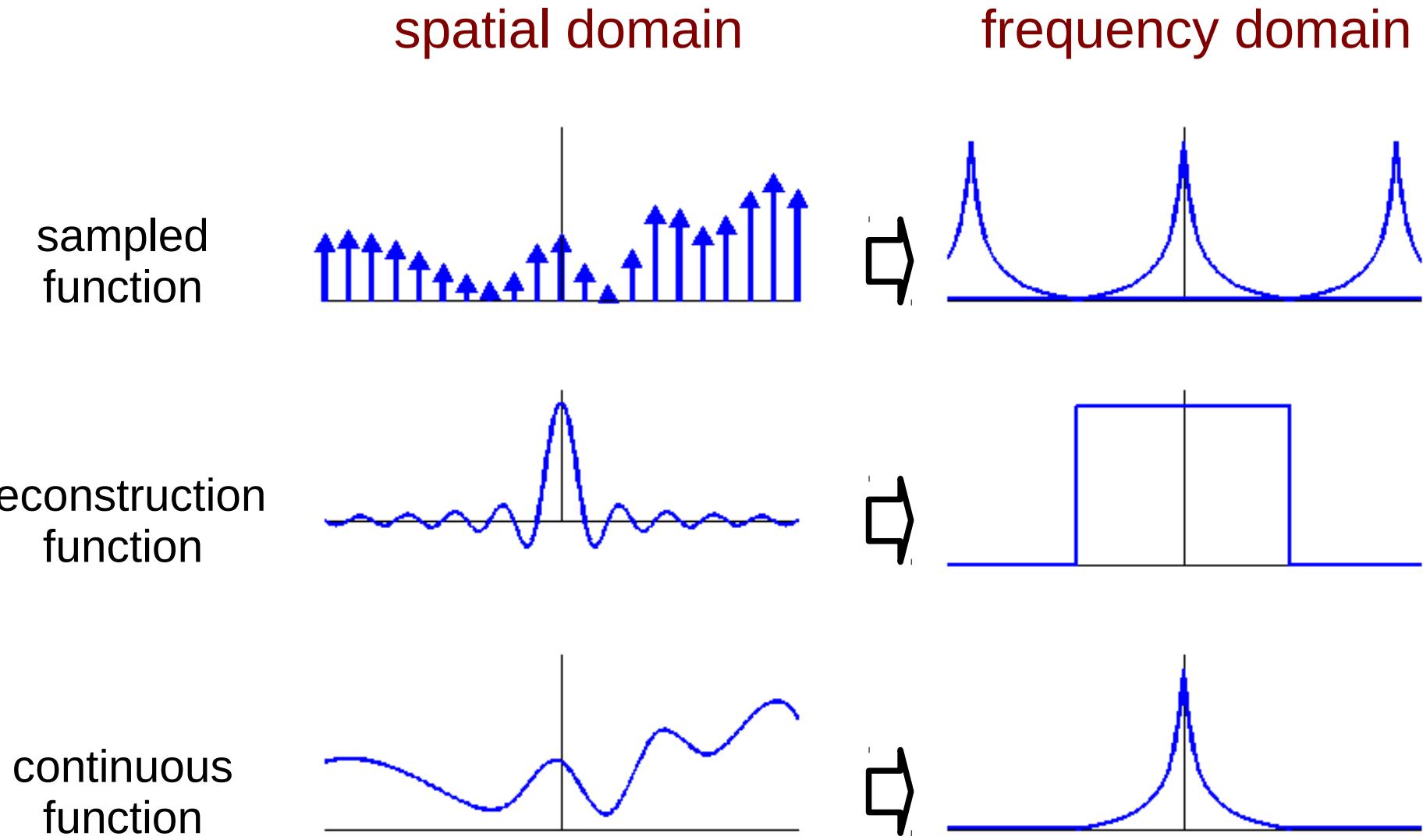


band limit  
(cutoff frequency)  
 $F(\omega) = 0, \omega > \omega_c$

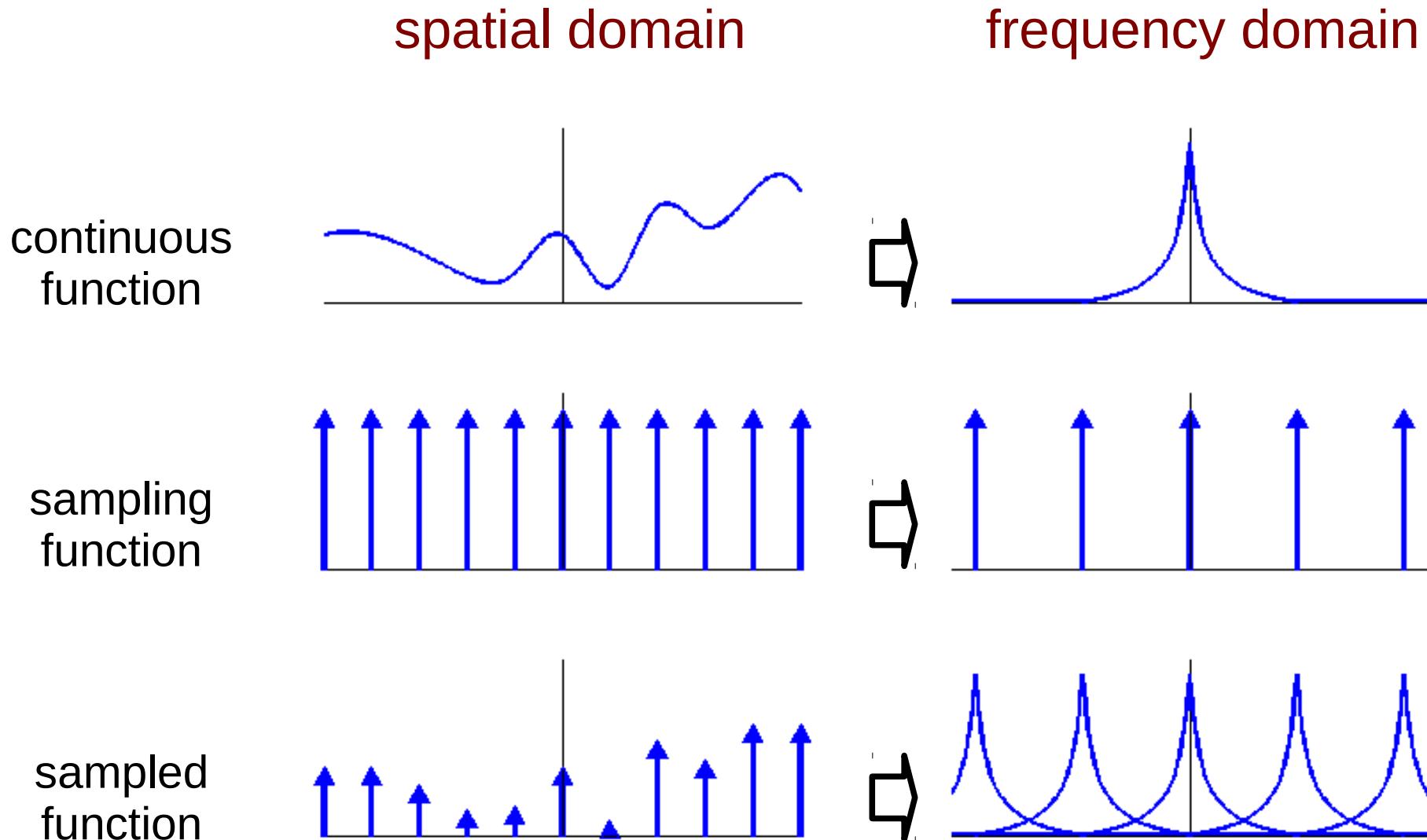
# Fourier analysis of sampling



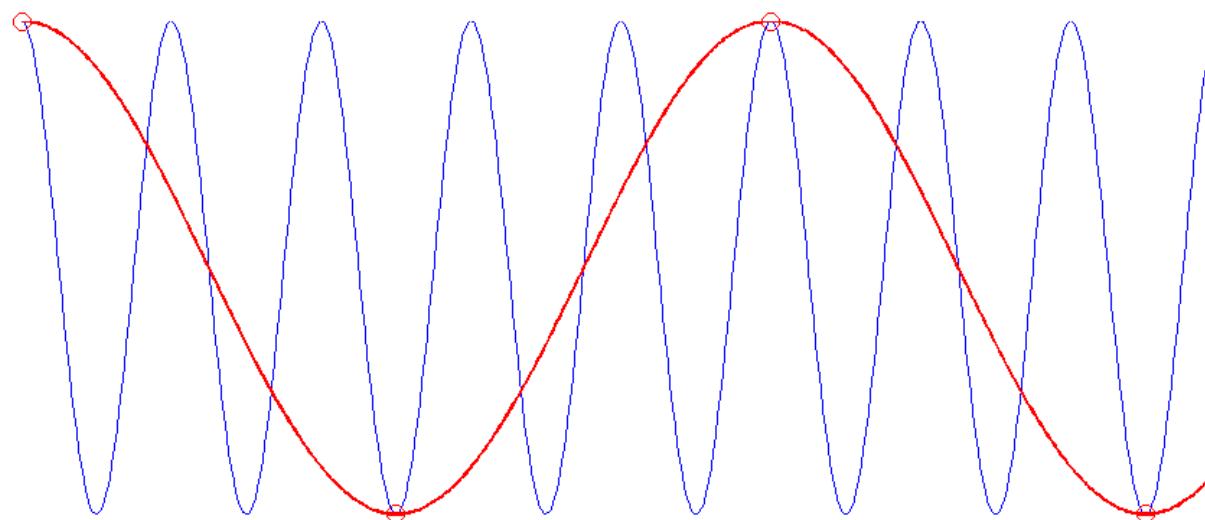
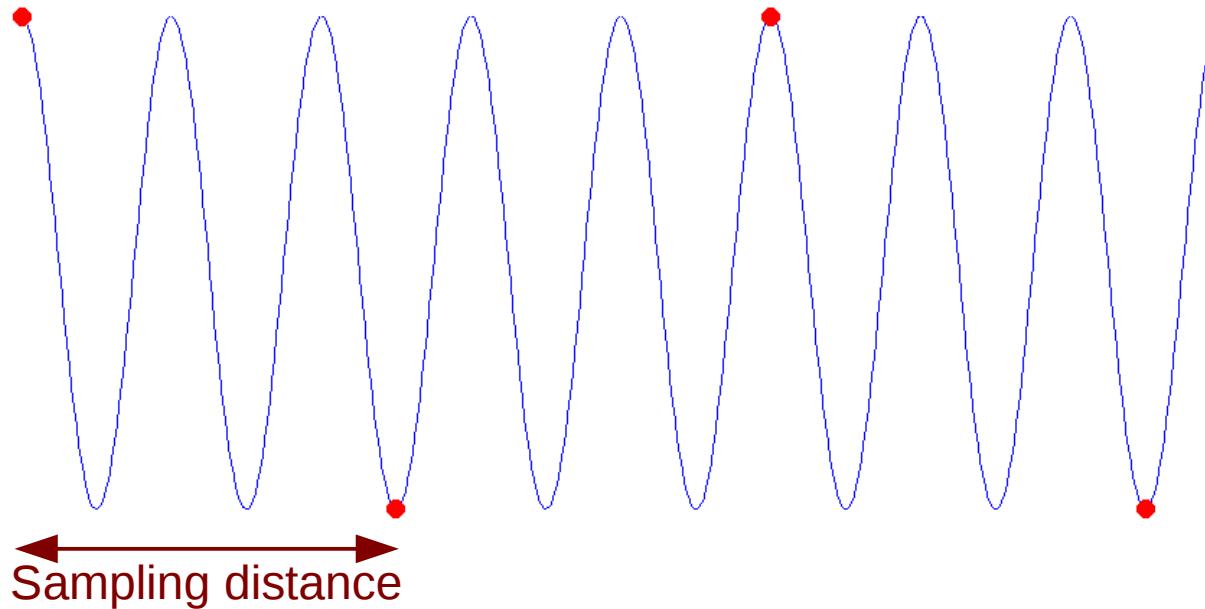
# Fourier analysis of interpolation



# Aliasing



# Aliasing



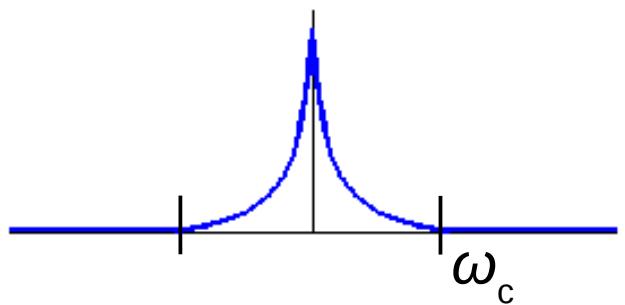
# Aliasing



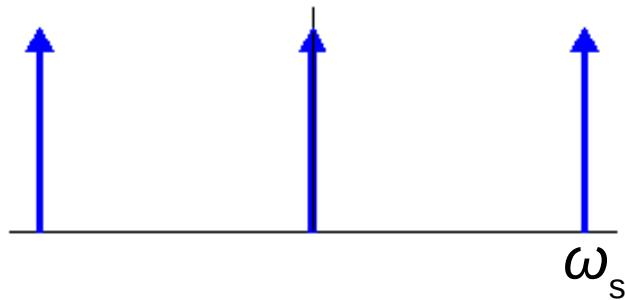
# Avoid aliasing

frequency domain

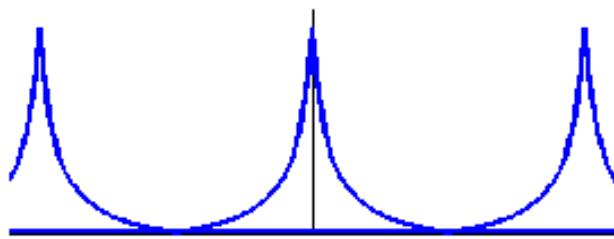
continuous  
function



sampling  
function



sampled  
function



$$F(\omega) = 0, \omega > \omega_c$$

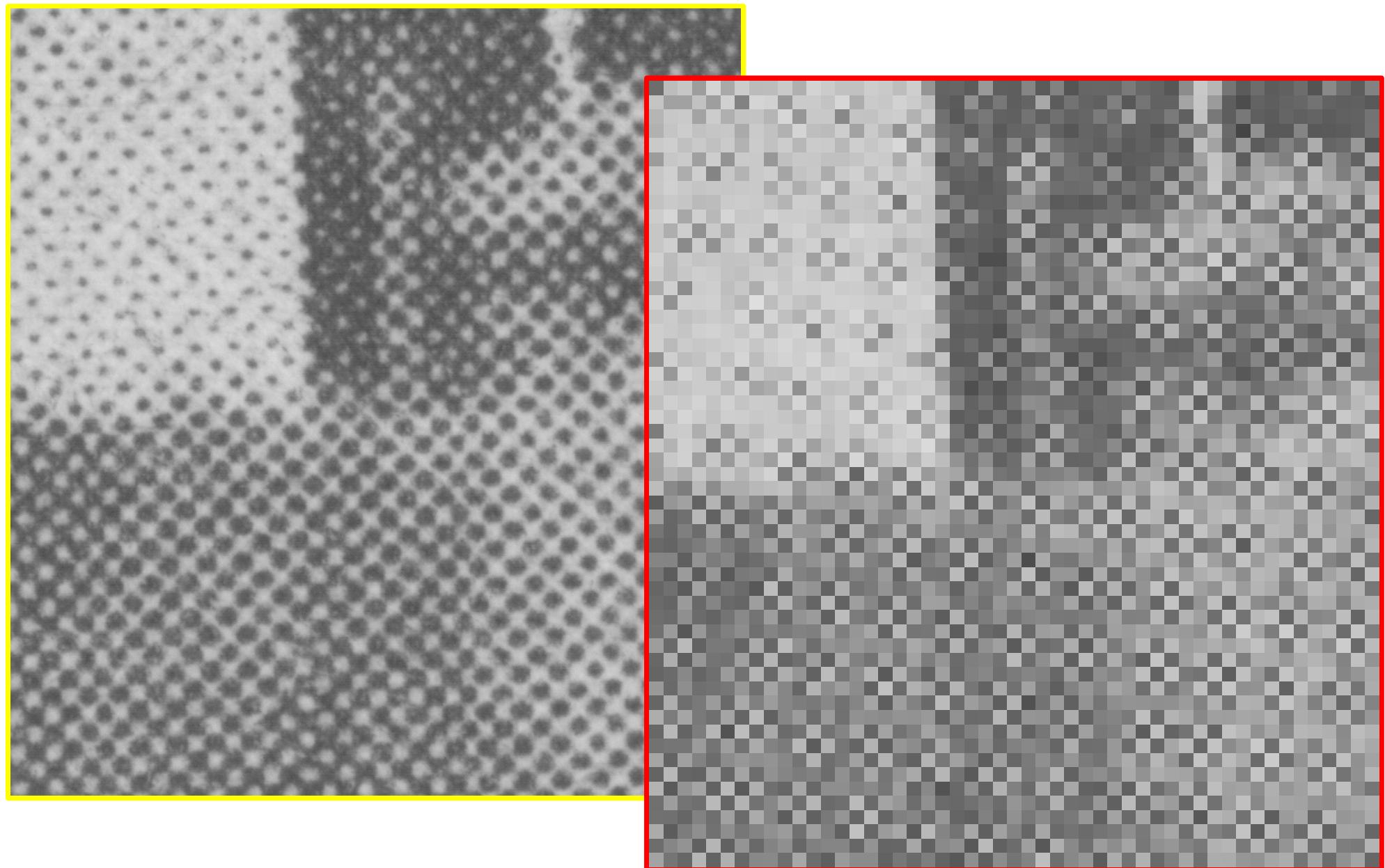
$$\omega_s > 2\omega_c$$

Minimum sampling frequency  
Nyquist frequency

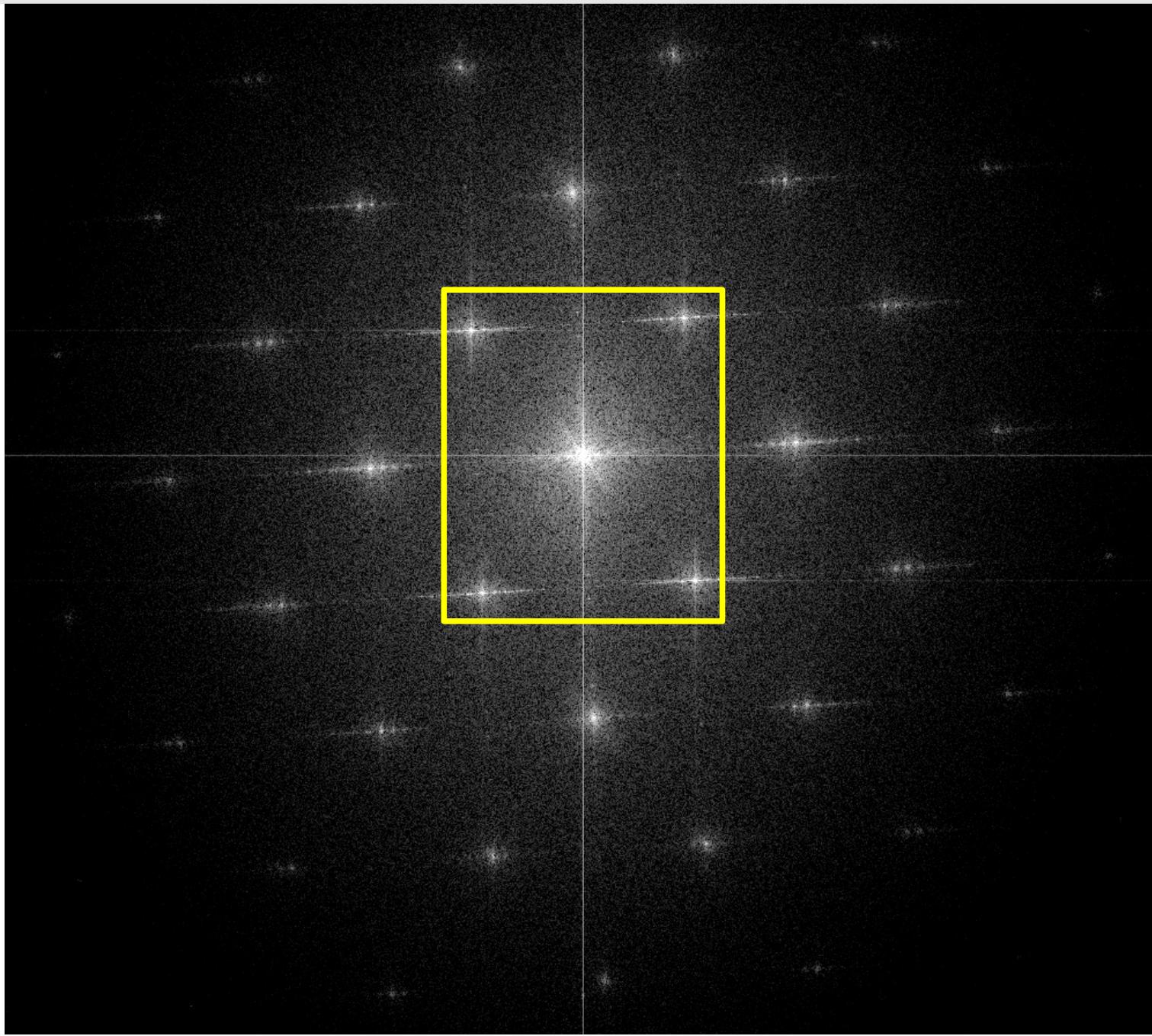
# Example: aliasing



# Example: aliasing

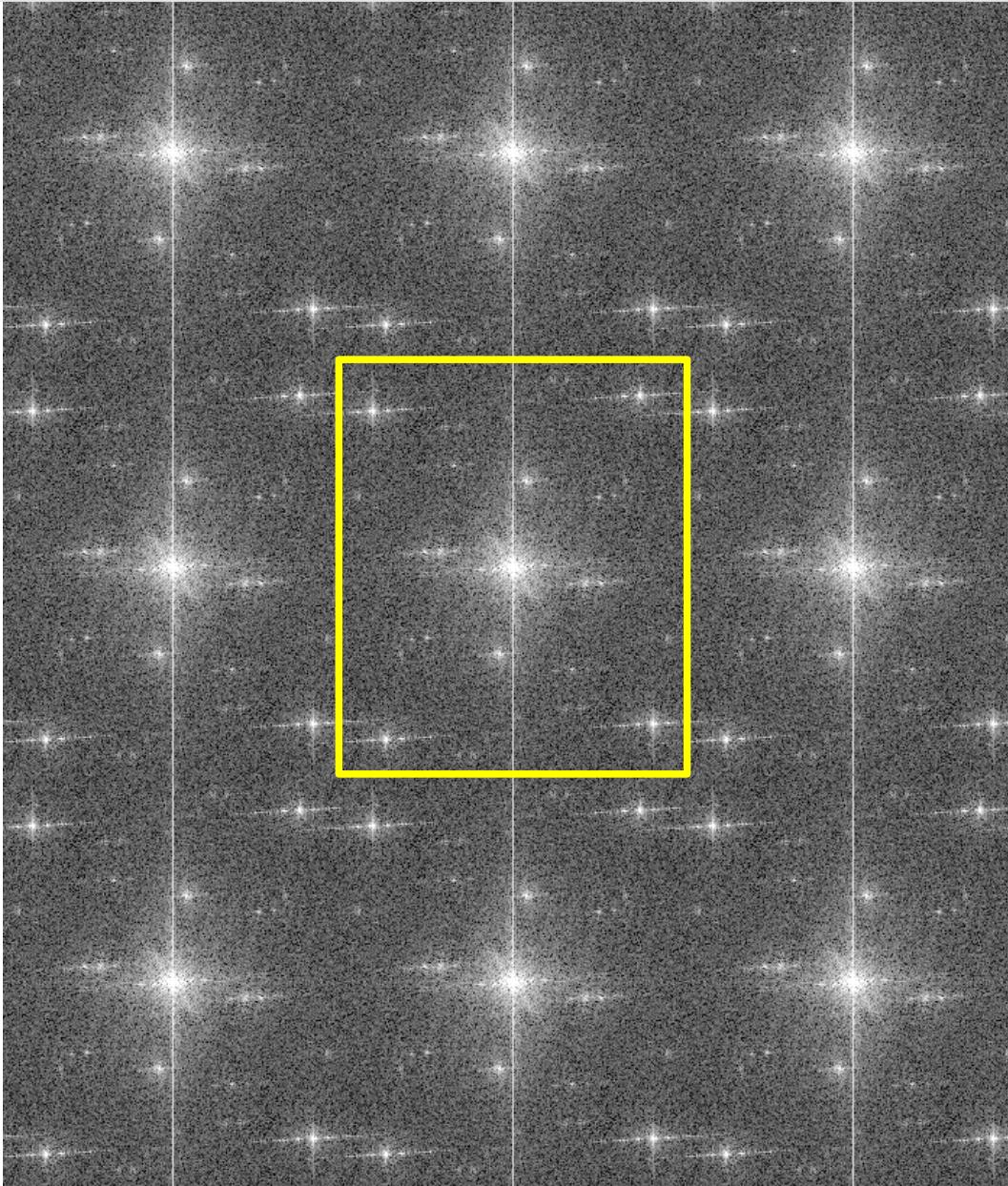


# Example: aliasing



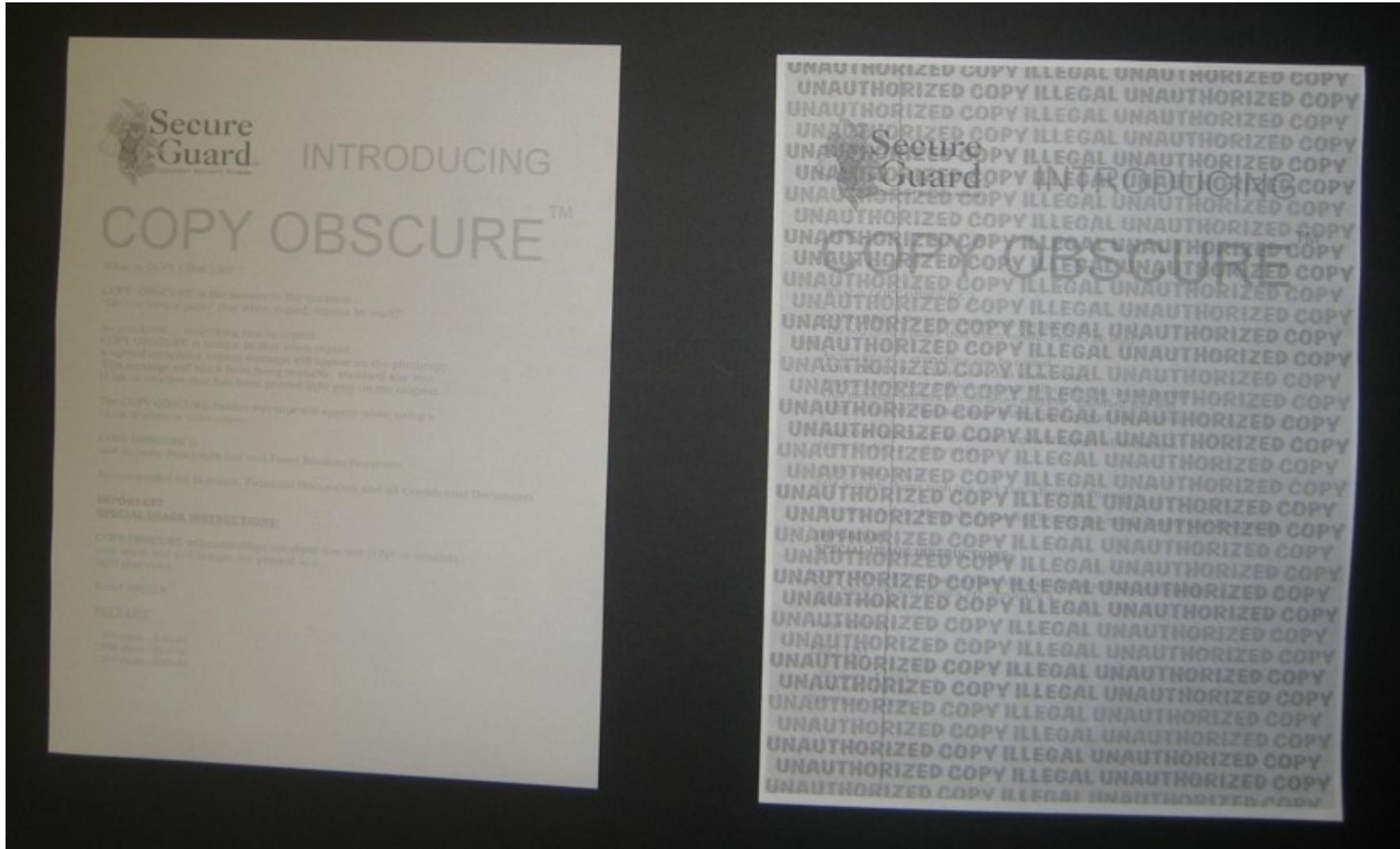
When we  
downsample,  
we only keep  
this part!

# Example: aliasing

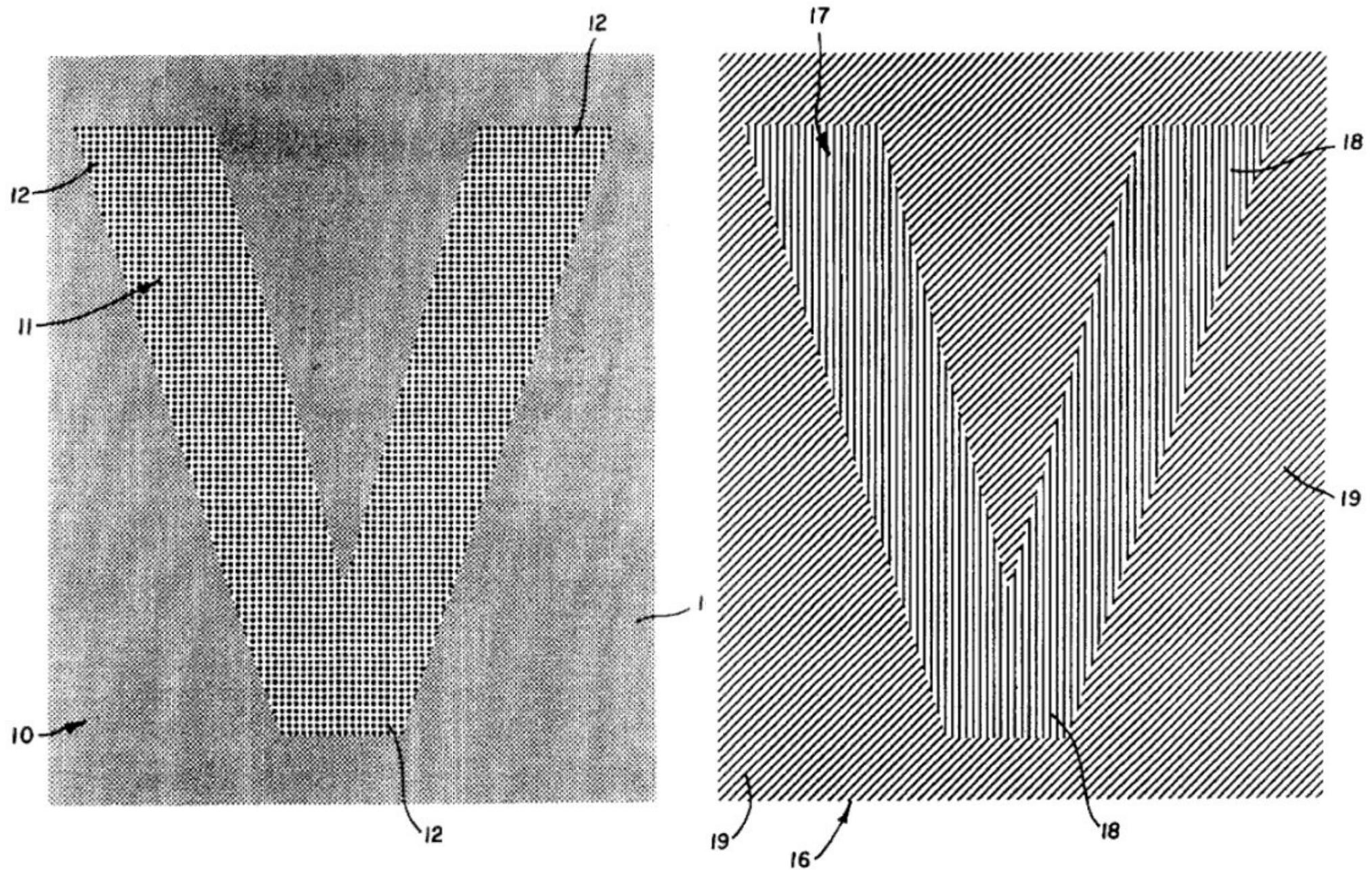


The spectrum is replicated, higher frequencies being duplicated as lower frequencies.

# Example: Moire



# Example: Moire



# Summary of today's lecture

- The Fourier transform
  - decomposes a function (image) into trigonometric basis functions (sines & cosines)
  - is used to analyse frequency components
  - is computed independently for each dimension
- The DFT can be computed efficiently through the FFT algorithm
- Convolution can be studied through the FT
  - and filters can be designed in the Fourier domain
  - $\mathcal{F}\{f \otimes h\} = \mathcal{F}\{f\} \cdot \mathcal{F}\{h\}$
- Aliasing can be understood through the FT

# Reading assignment

- The Fourier transform and the DFT
  - Sections 4.2, 4.4, 4.5, 4.6, 4.11.1
- Filtering in the Fourier domain
  - Sections 4.7, 4.8, 4.9, 4.10, 5.4
- Sampling and aliasing
  - Sections 4.3, 4.5.4
- The FFT
  - Section 4.11.3
- Exercises:
  - 4.14, 4.21, 4.22, 4.42, 4.43
  - 4.27, 4.29

(feel free to solve these in MATLAB)

