Summary of last week's lecture

- 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.
- Aliasing can be understood through the Fourier transform
- FFT is a fast algorithm to compute the DFT
- 2D FT is separable

Fourier transform

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

$f(x)$

$=$

$F(\omega_1)e^{i\omega_1 x}$

$+$

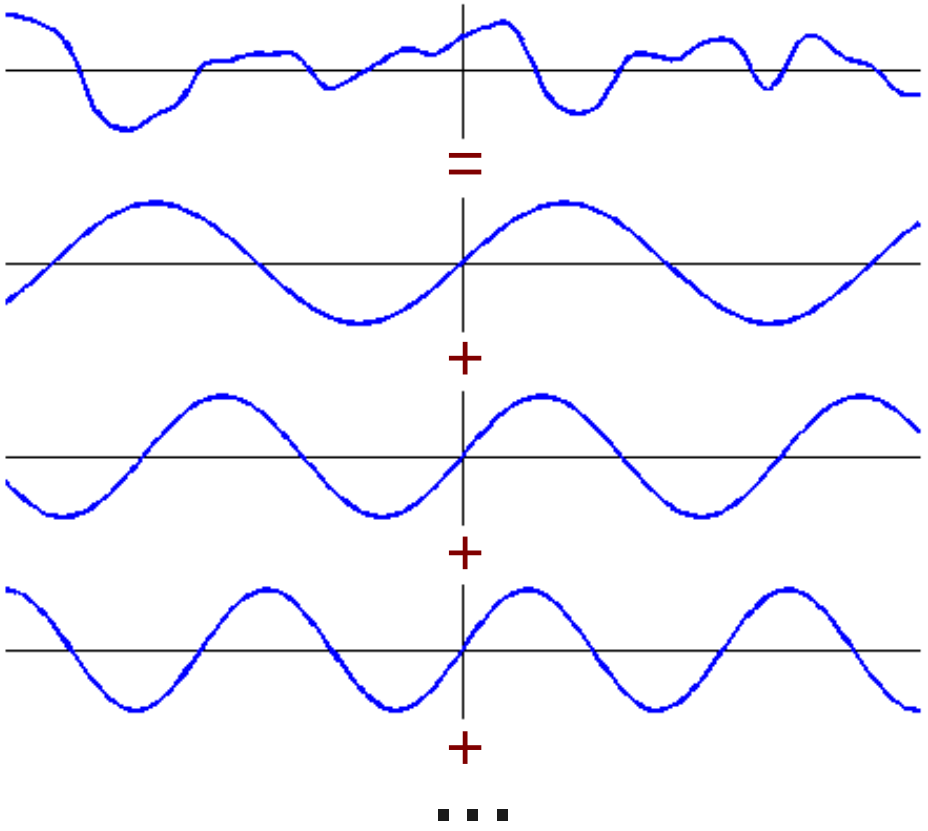
$F(\omega_2)e^{i\omega_2 x}$

$+$

$F(\omega_3)e^{i\omega_3 x}$

$+$

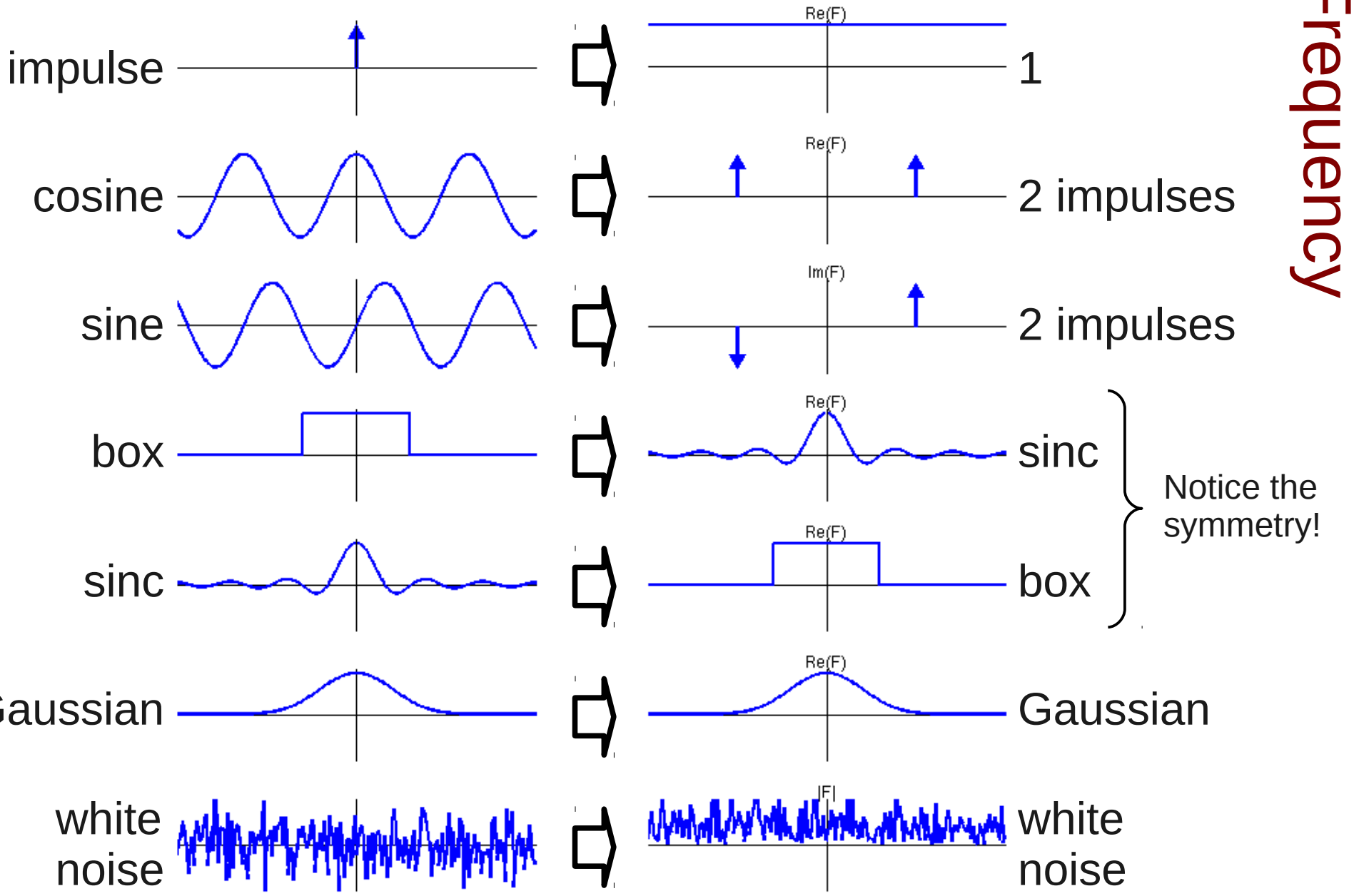
$\ldots$



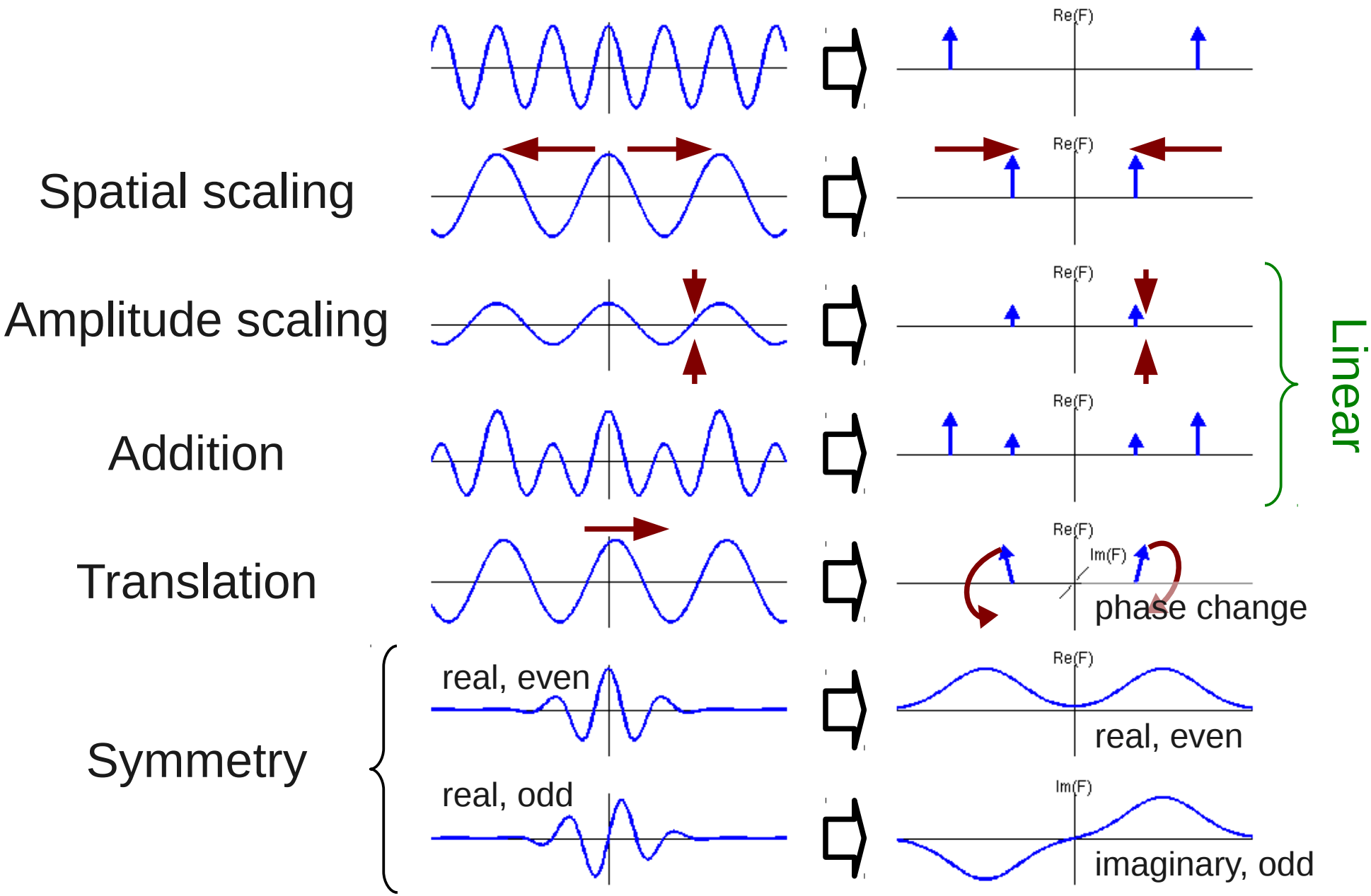
Fourier transform pairs

Spatial

Frequency

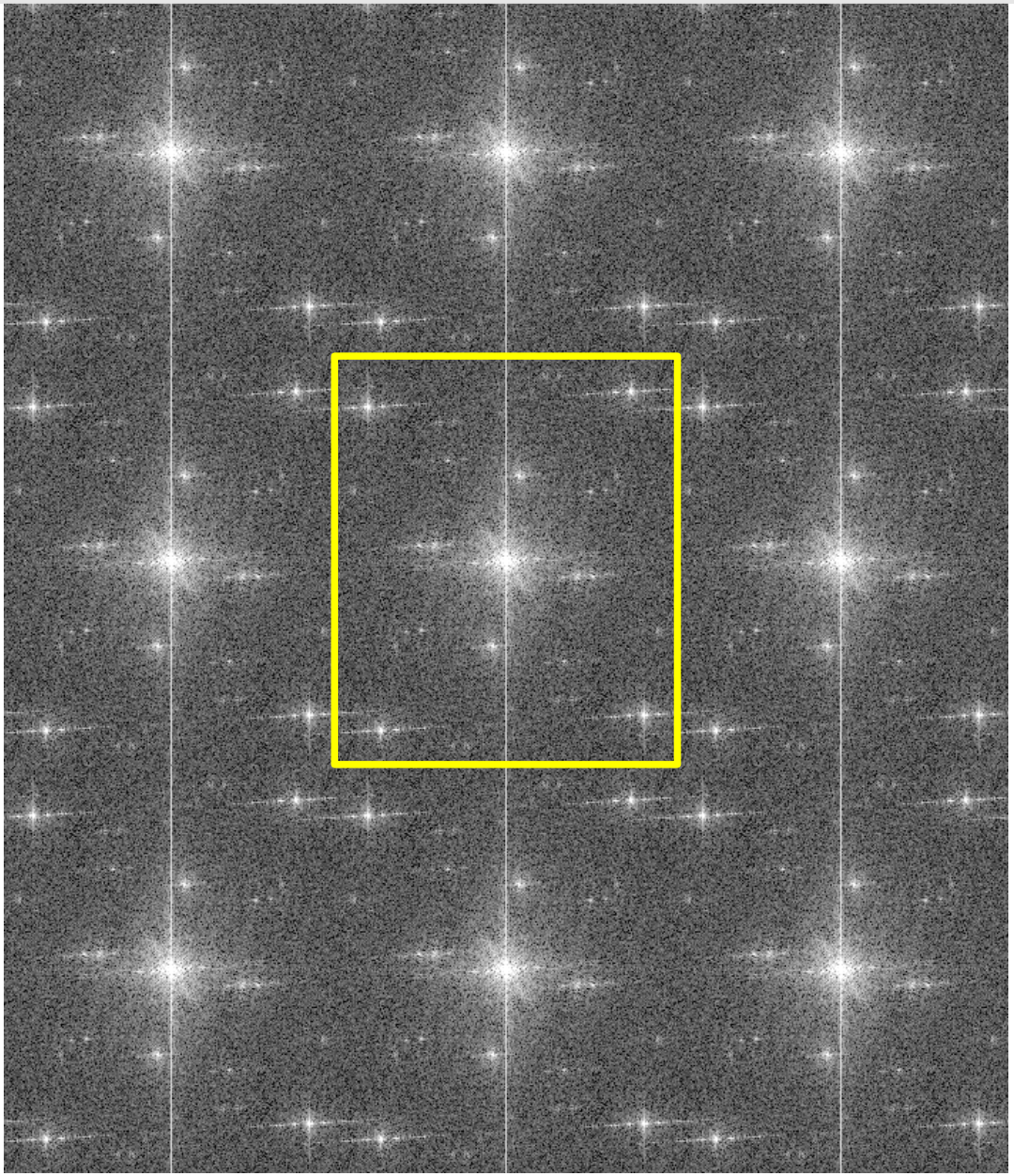


Properties of the Fourier transform



Fourier example

Example: aliasing



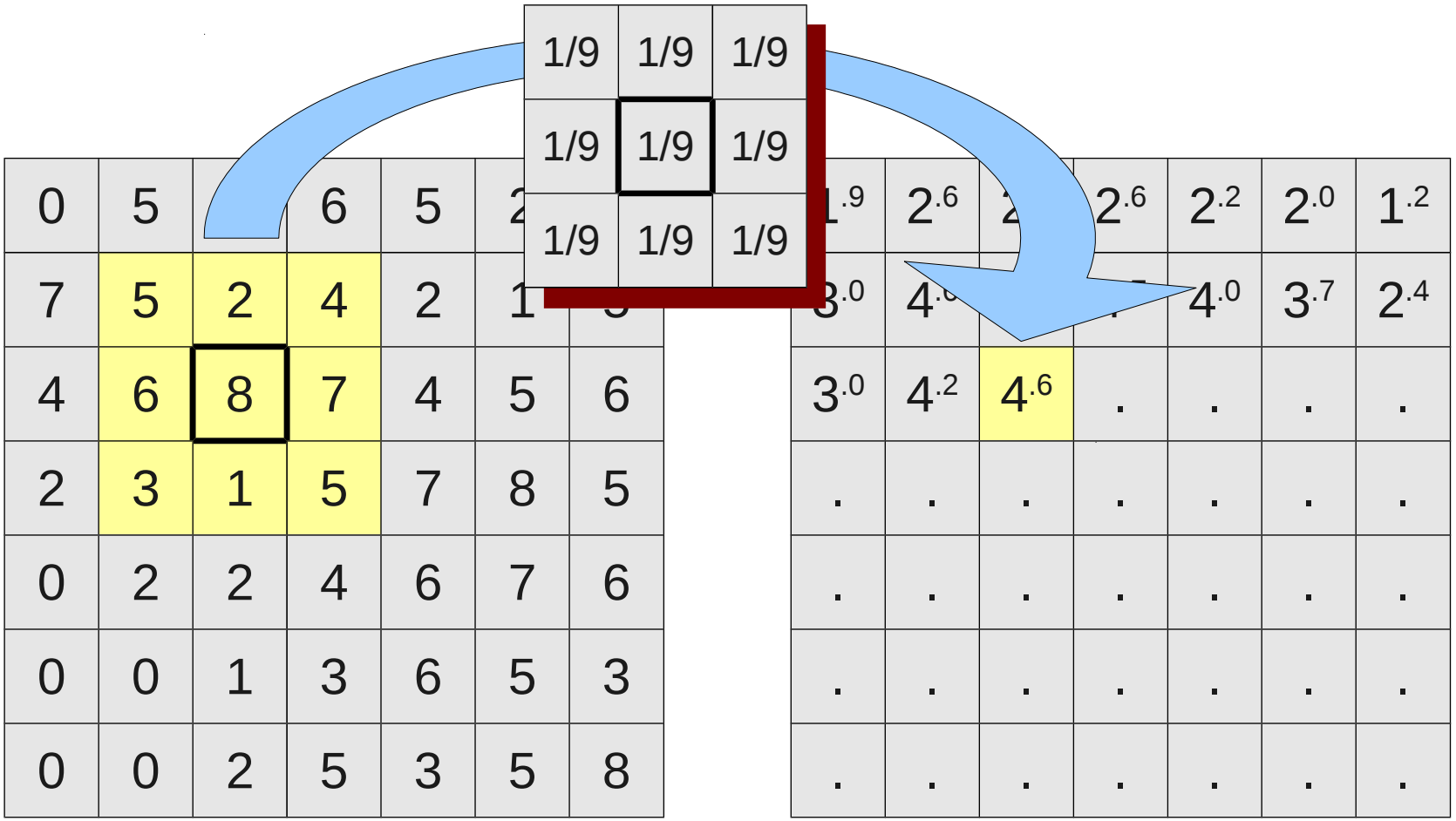
The spectrum is replicated, higher frequencies being duplicated as lower frequencies.

Today's lecture

- Convolution
  - What is it?
  - What is it useful for?
  - How can I compute it?
- How is interpolation related to convolution?
- Filtering in the Fourier Domain
  - High-pass, low-pass, band-pass filtering
  - Inverse filtering
  - The Wiener filter

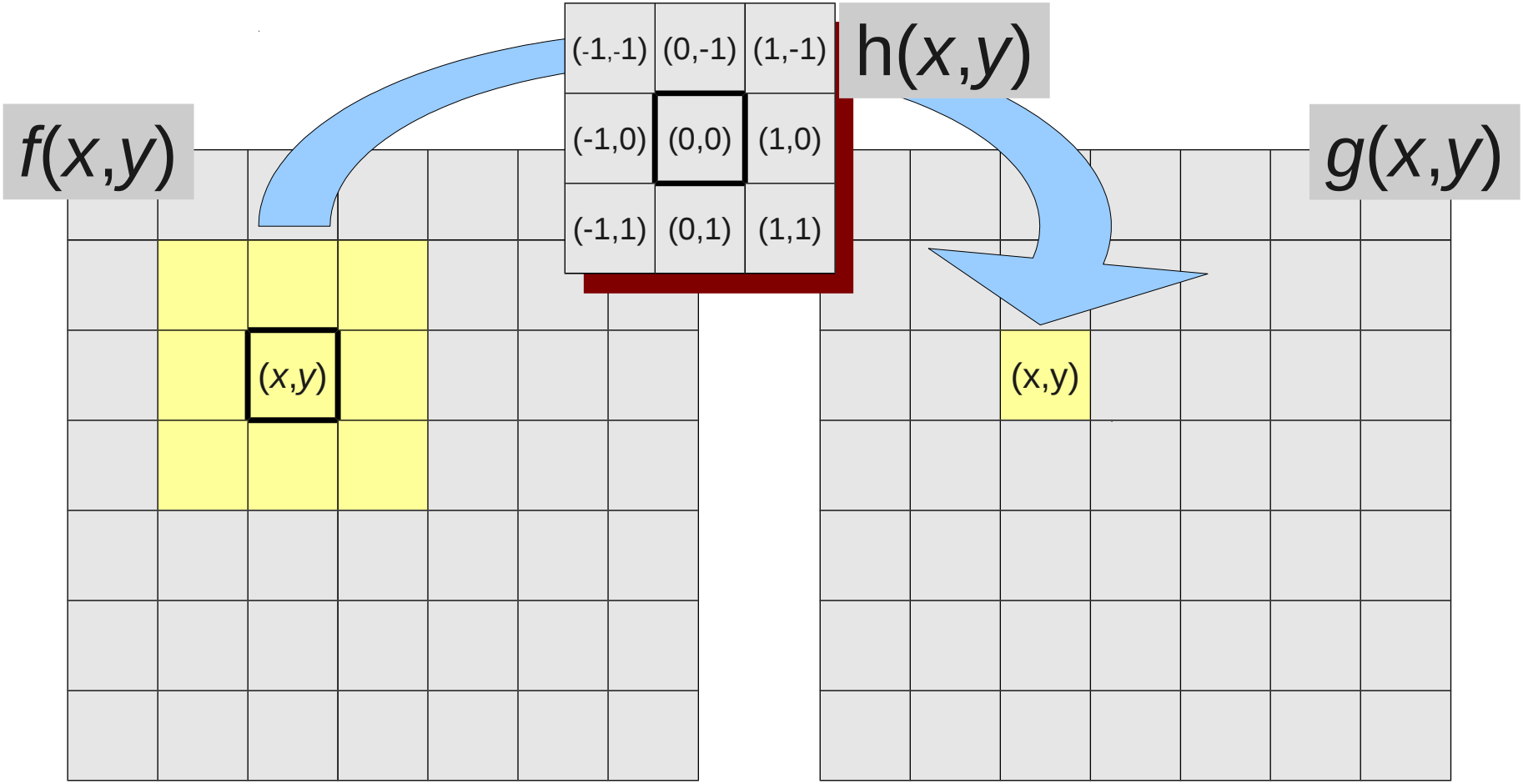
Linear neighbourhood operation

- For each output pixel, take the weighted sum of the input values of the pixels from its neighbourhood.



Linear neighbourhood operation

- For each output pixel, take the weighted sum of the input values of the pixels from its neighbourhood.



Convolution

$$g(t) = f(t) \otimes h(t)$$

h is:
- impulse response function
- point-spread function
- convolution kernel

$$g(t) = \int_{-\infty}^{\infty} f(\tau)\, h(t-\tau)\, d\tau$$

$$g[n] = \sum_{k=0}^{N-1} f[k]\, h[n-k]$$

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: $shift(f) \otimes h = shift(f \otimes h)$
  (= shift invariant)
- Commutative: $f \otimes h = h \otimes f$
- Associative: $f \otimes (h_1 \otimes h_2) = (f \otimes h_1) \otimes h_2$

Convolution in the Fourier domain

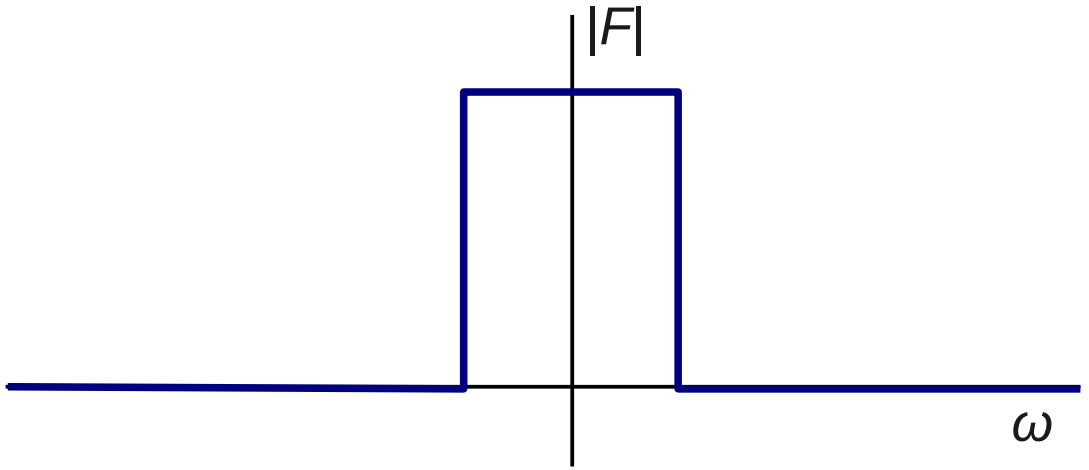
- The Convolution property of the Fourier transform:

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

- Thus we can calculate the convolution through:
  - F = FFT(f)
  - H = FFT(h)
  - $G = F \cdot H$
  - g = 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!

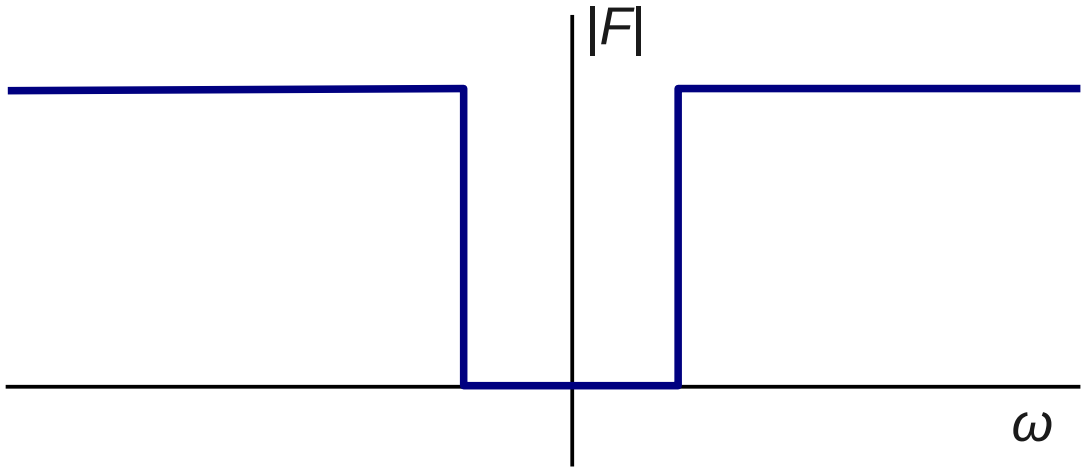
Low-pass filtering

- Linear smoothing filters you saw in an earlier lecture are all low-pass filters.
  - Uniform
  - Butterworth
  - Gauss
- Low-pass means low frequencies are not altered, high frequencies are attenuated



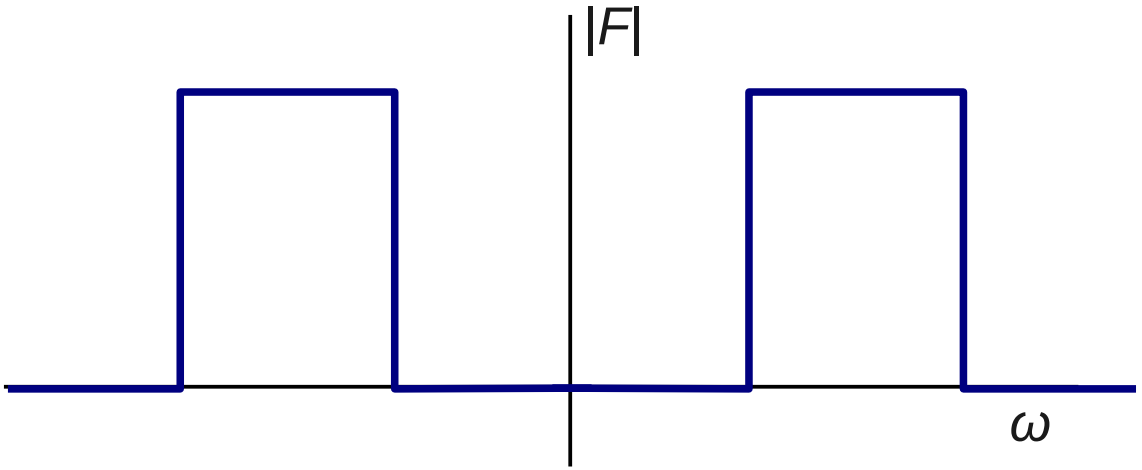
High-pass filtering

- The opposite of low-pass filtering: low frequencies are attenuated, high frequencies are not altered



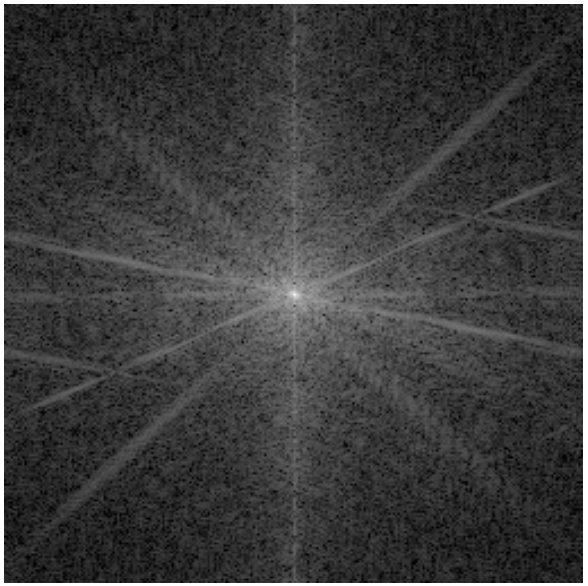
Band-pass filtering

- 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).



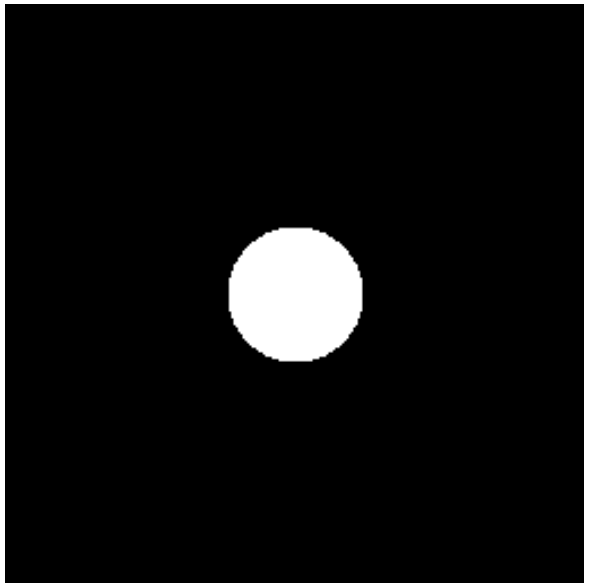
Example: low-pass filtering

input image *f*

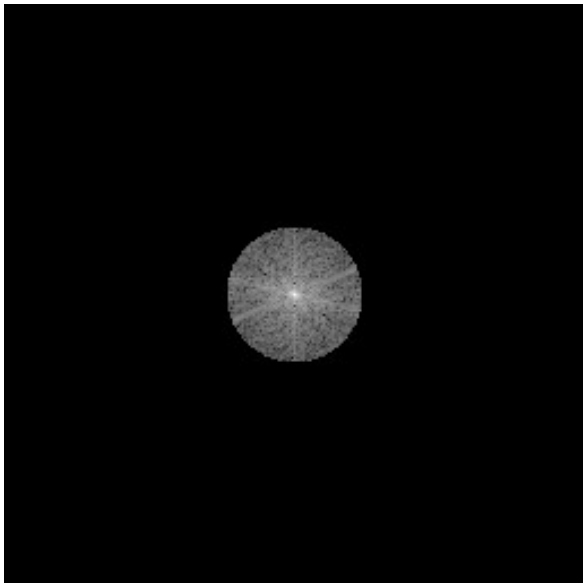


Fourier transform *F*

Example: frequency domain filtering



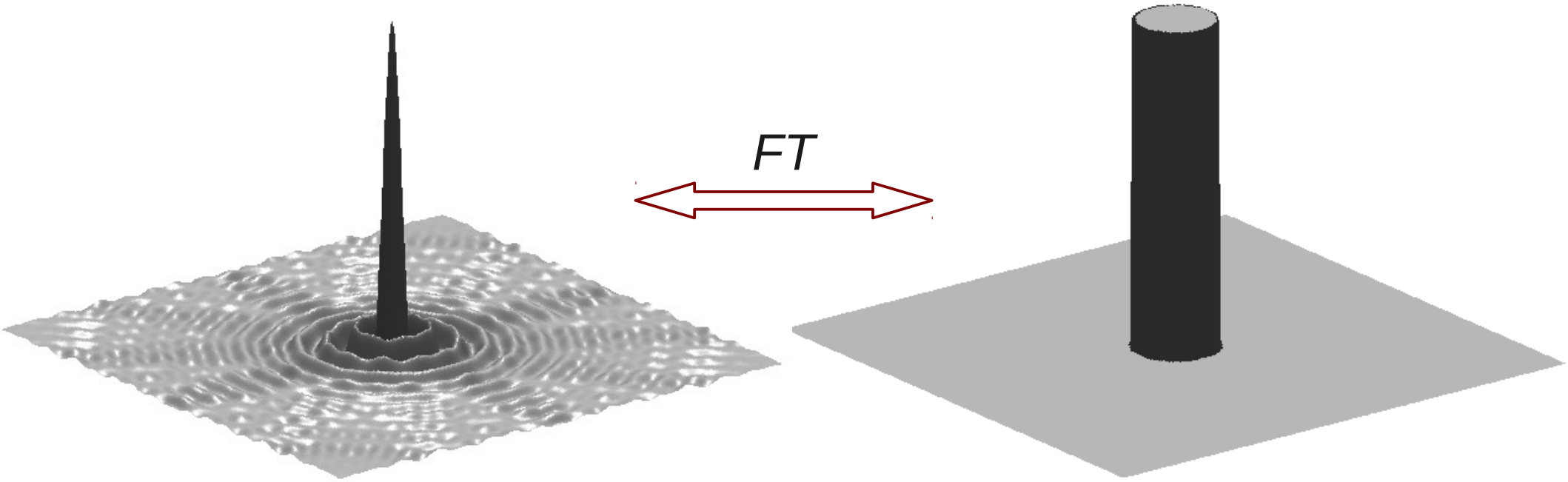
Fourier filter *H*



$G = F \ H$

filtered image *g*

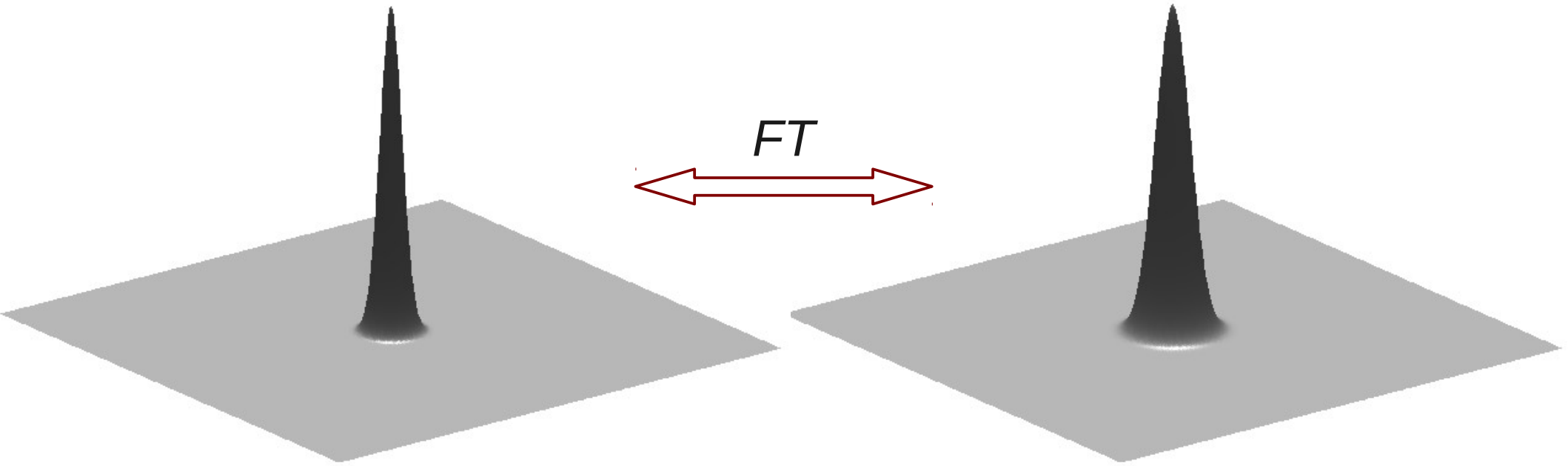
Why the ringing?



Spatial domain

Frequency domain

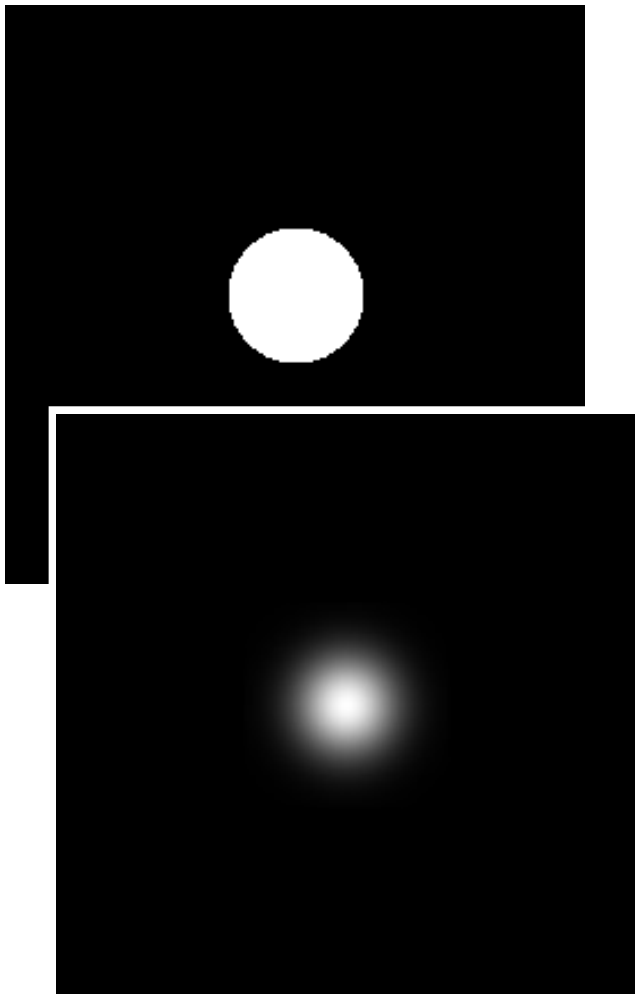
What is the solution?



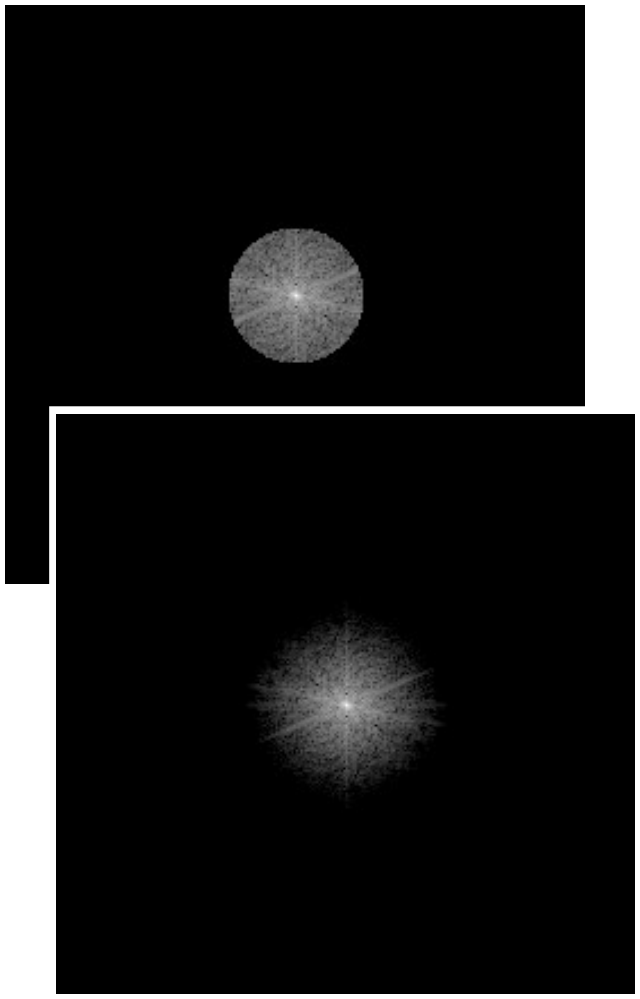
Spatial domain

Frequency domain

Example: frequency domain filtering



Fourier filter *H*



$G = F \ H$

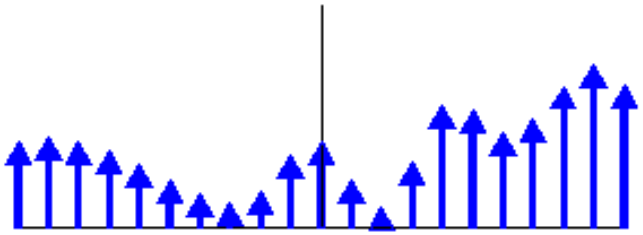
filtered image *g*

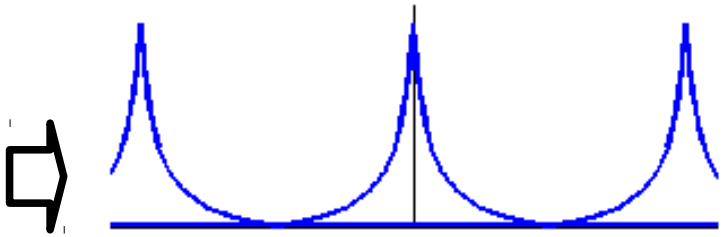
Reconstruction

spatial domain

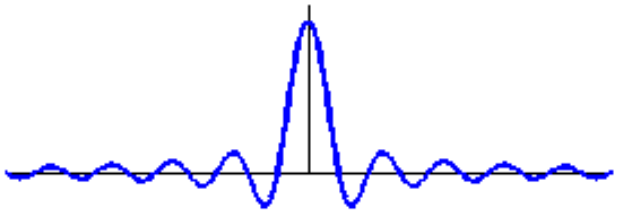
frequency domain

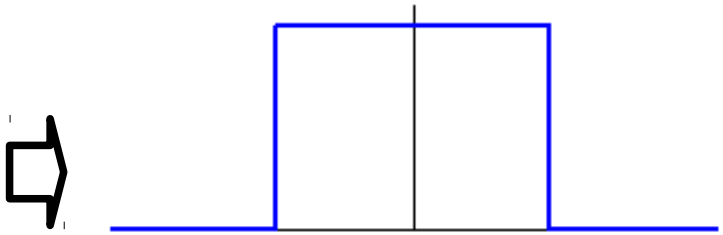
sampled function



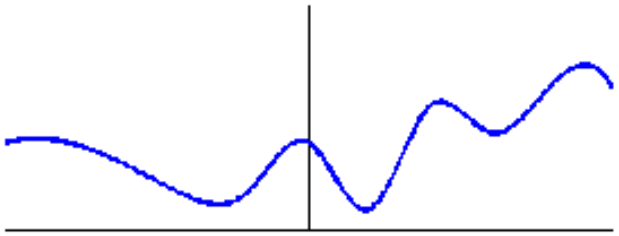


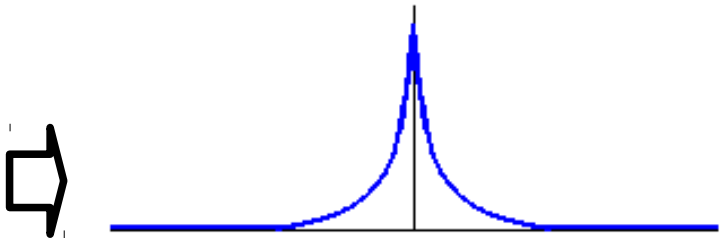
reconstruction function



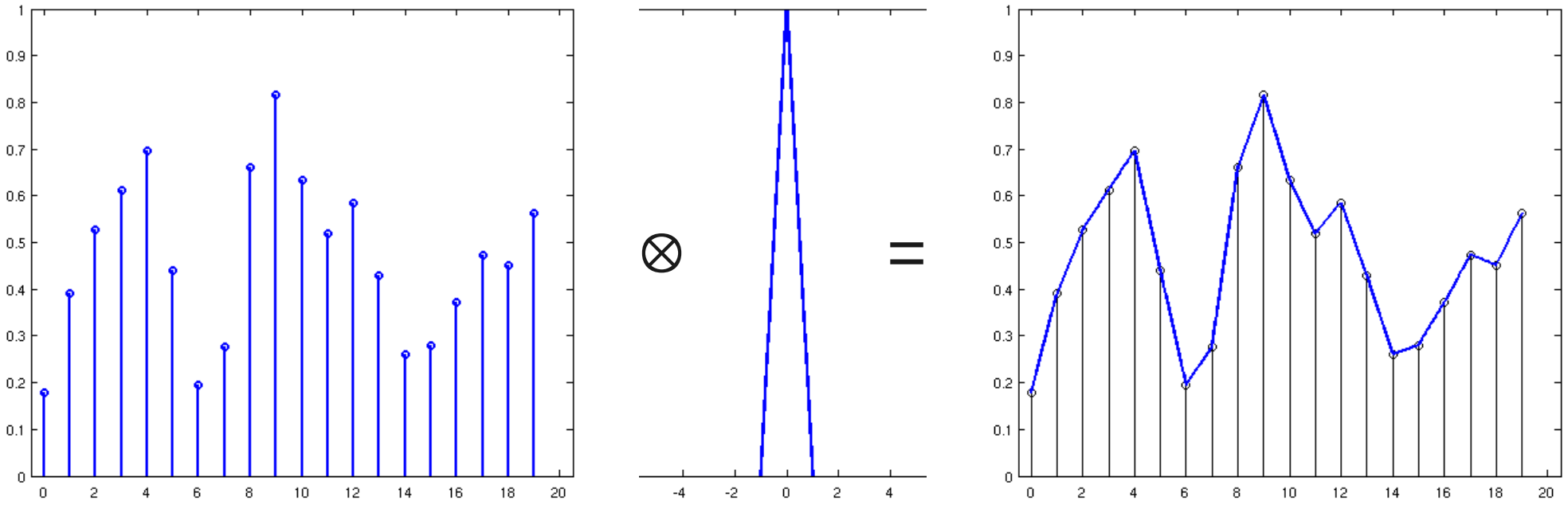


continuous function



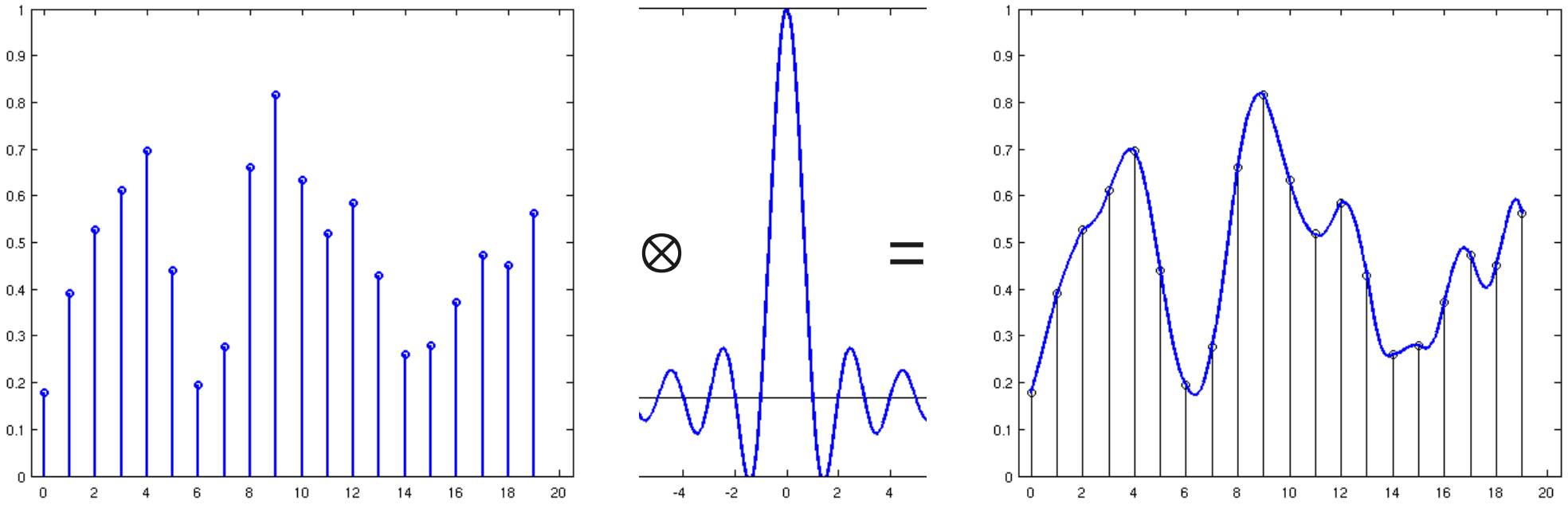


Reconstruction revisited



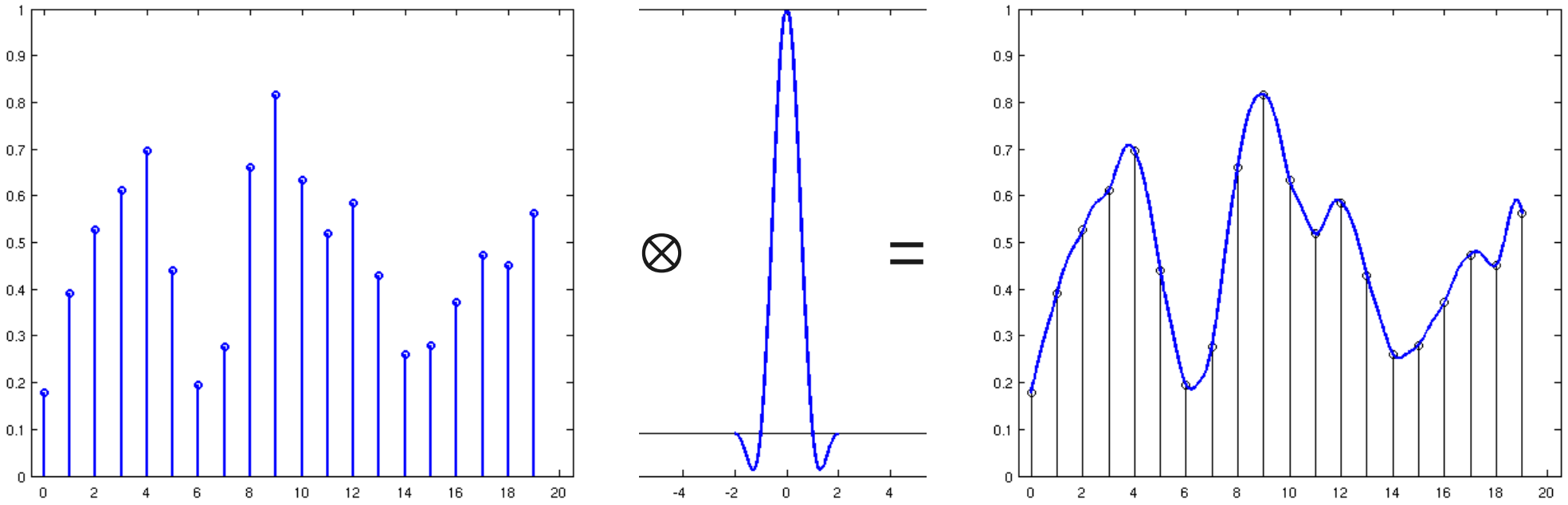
Linear interpolation is a convolution with a triangular kernel

Reconstruction revisited



Ideal interpolation is a convolution with a sinc kernel

Reconstruction revisited



Lanczos kernel is a windowed sinc kernel

Reconstruction revisited

Interpolation kernels in frequency domain

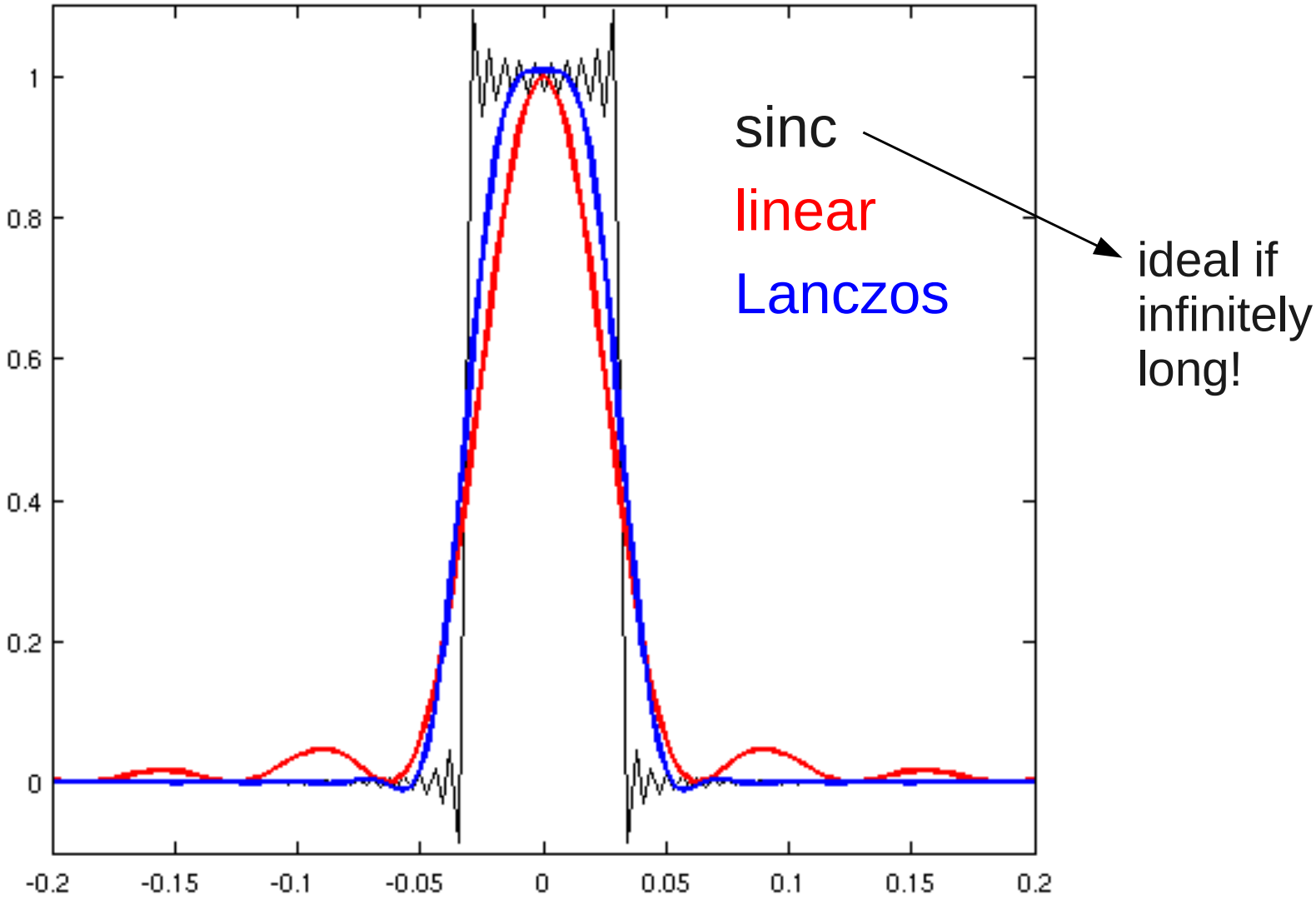


Image degradations

- Degradations modeled as a convolution:
  - out-of-focus image (low-pass filter)
  - motion blur (kernel reflects path of motion)
- Non-linear degradations:
  - optical distortion: barrel/pincushion, chromatic aberration, ...
  - varying focal depth
- Noise
  - additive (Gaussian) or multiplicative (Poisson)
  - correlated (pink) or uncorrelated (white)
  - weird noise types: impulse (salt & pepper), structured, ...
  - many possible sources...

Inverse filtering

$$g = f \otimes h \;\Rightarrow\; G = F\ H$$

$$\hat{F} = \frac{G}{H}$$

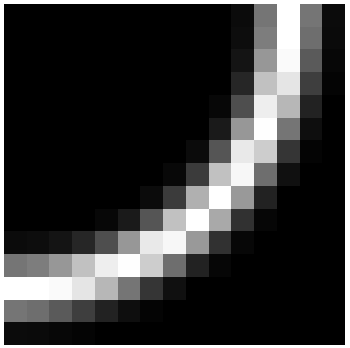
$$h' = \mathrm{IFFT}\left(\frac{1}{H}\right)$$

$$\hat{f} = g \otimes h'$$

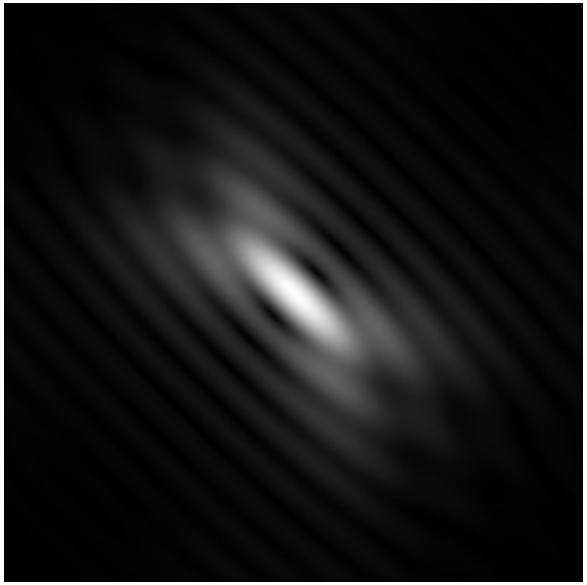
Inverse filtering is also known as deconvolution

Inverse filtering

Image with motion blur (simulated!)



Kernel used to simulate motion



FFT of kernel

Inverse filtering

Image with motion blur

$$\hat{F} = \frac{G}{H}$$

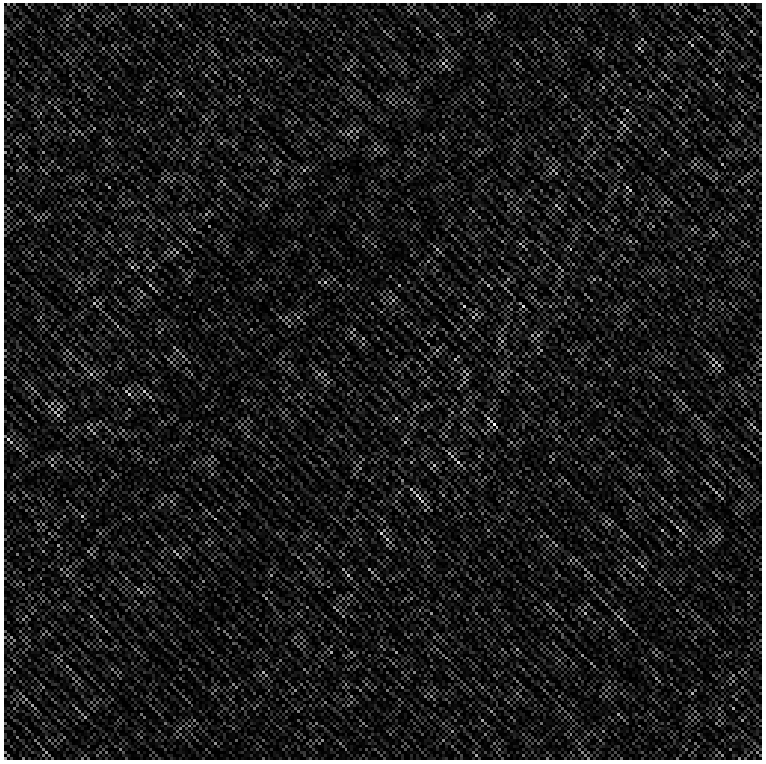
Inverse filtering

Image with motion blur
and a tiny bit of noise

$$\hat{F} = \frac{G}{H}$$

Inverse filtering

Image with motion blur and more noise



$$\hat{F} = \frac{G}{H}$$

Inverse filtering

Image with motion blur and more noise

Wiener filter

The problem with inverse filtering

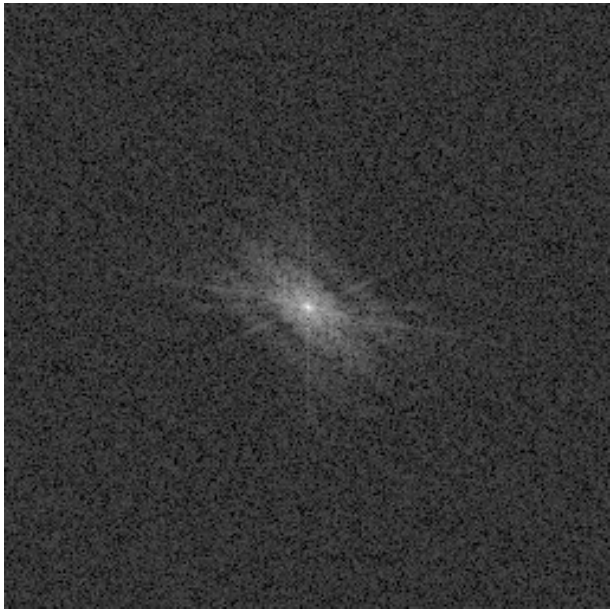
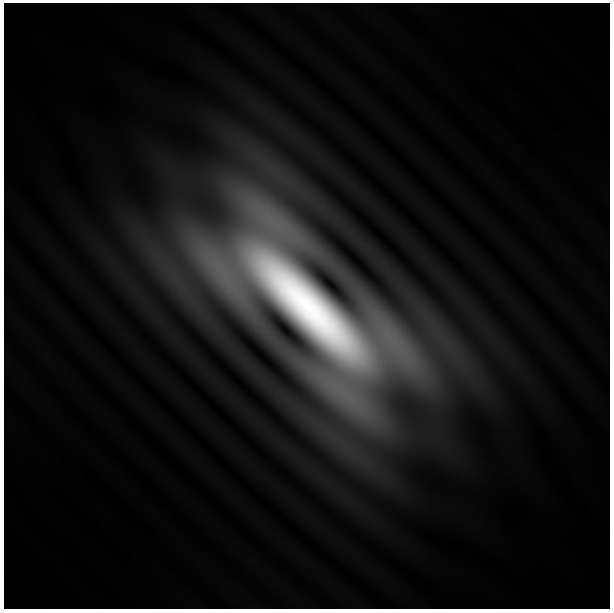
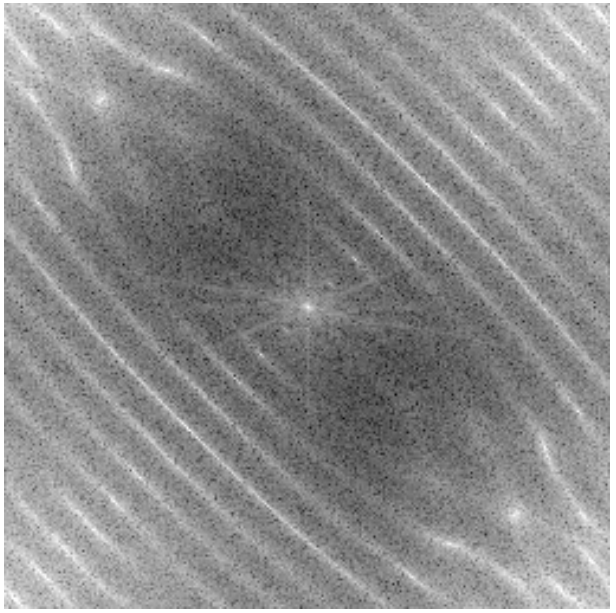


Image with motion blur and noise



Motion blur convolution kernel



Result of inverse filtering

Wiener filter

Solution to mean square error problem: $\sum \left(\hat{f} - f\right)^2$

$$\hat{F} = \frac{1}{H} \frac{\|H\|^2}{\|H\|^2 + \frac{P_n}{P_s}} G$$

Requires knowledge of power spectrum of *f* and noise

Assumes noise is additive, not correlated to the image

$$\hat{F} = \frac{1}{H} \frac{\|H\|^2}{\|H\|^2 + K} G$$

Inverse filtering

Image with motion blur and more noise

Wiener filter, with knowledge of power spectrum of undegraded image and noise

Inverse filtering

Image with motion blur
and more noise

Wiener filter,
after a good guess
of parameter K

The problem with inverse filtering

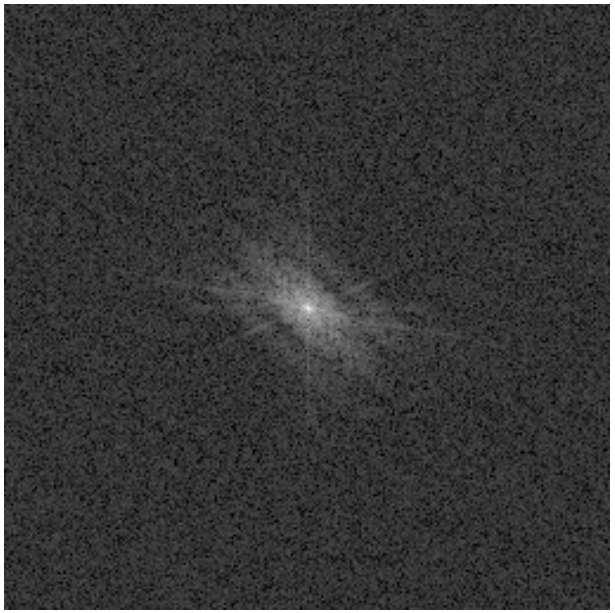
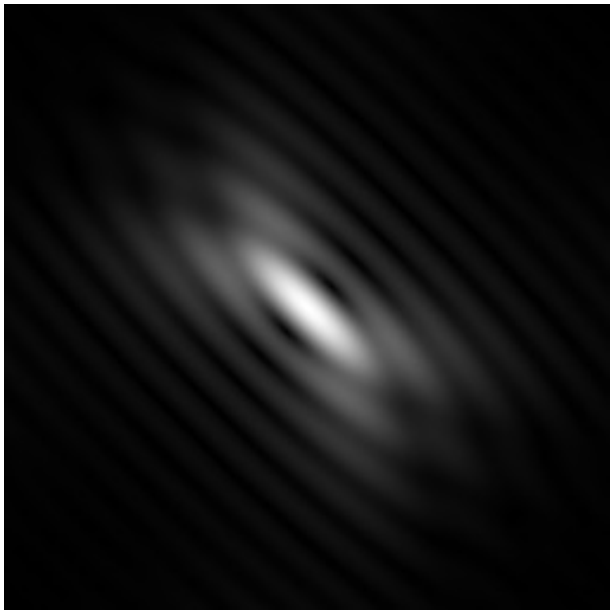
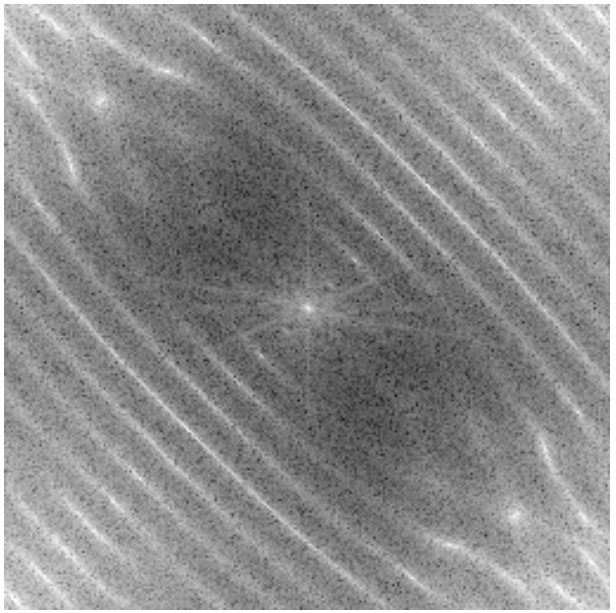


Image with motion blur and noise



Motion blur convolution kernel



Result of inverse filtering

The problem with inverse filtering

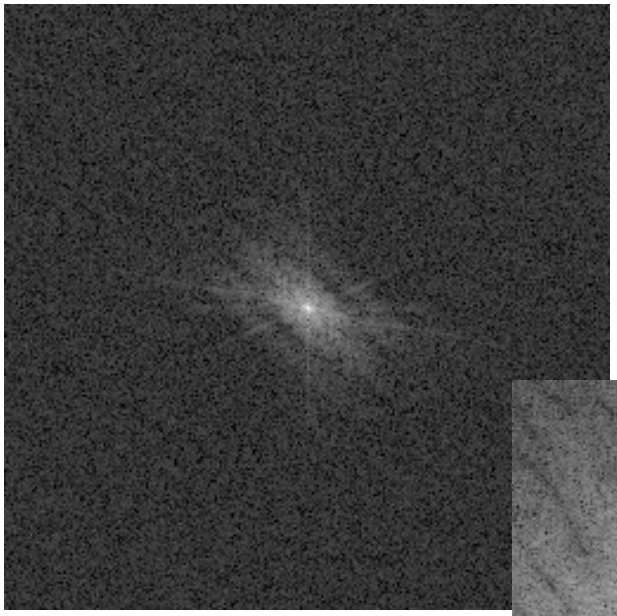
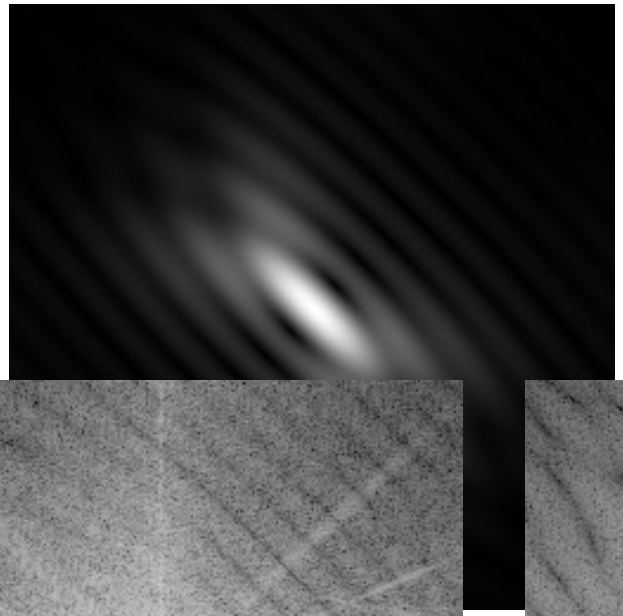
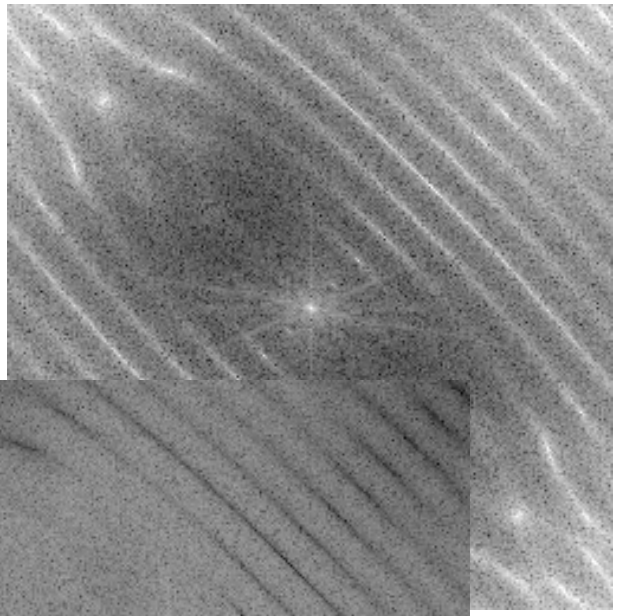


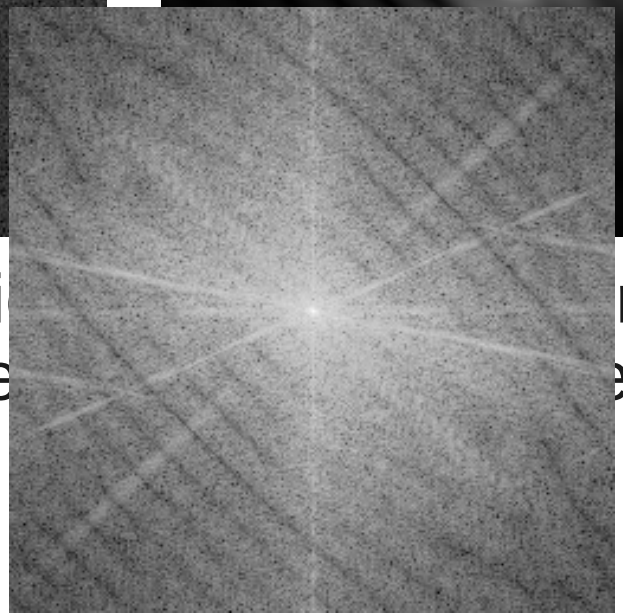
Image with moti
blur and noise



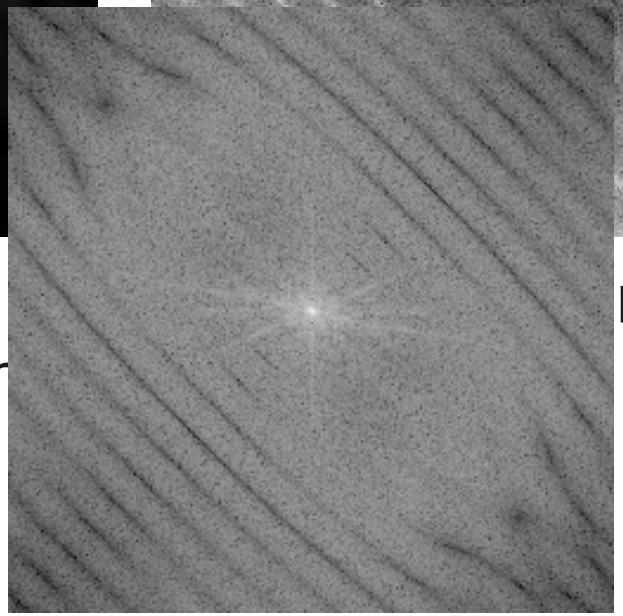
r
err



rse



Wiener filter,
with knowledge of
power spectra



Wiener filter,
after a good guess
of parameter K

Iterative deconvolution

Iteratively refine the estimate $\hat{f}$ such that convolving it with h gives the input image g

$$g = \hat{f} \otimes h + n$$

One way is to define an energy function to minimize
(minimization of such a function is standard technique)

$$E = \sum \left(g - \hat{f} \otimes h \right)^2 + \underbrace{\lambda \sum |\nabla \hat{f}|^2}$$

Requires a regularization: assume $\hat{f}$ is smooth

Estimating the PSF

$$g = f \otimes h$$

h is the convolution kernel, also known as:
- impulse response function
- point spread function (PSF)

To learn h, simply put $f(t) = \delta(t)$ into the system, then the output $g(t) = h(t)$

$$\delta(t) = 0\,, \quad t \neq 0$$

$$\int_{-\infty}^{\infty} \delta(t)\,\mathrm{d}t = 1$$

$$\delta[n] = \begin{cases} 1\,, & n = 0 \\ 0\,, & n \neq 0 \end{cases}$$

Estimating the PSF

- A tiny bright point can be the $\delta(t)$ to characterize a camera, but is difficult to find
- More common are sharp edges:

$$u(t) = \int \delta(t)\,\mathrm{d}t$$

$$u(t) \otimes h(t) = \int h(t)\,\mathrm{d}t$$

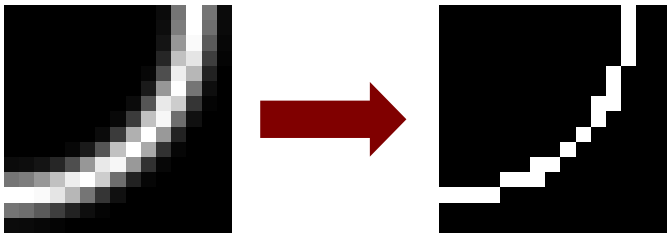
$$h(t) = \frac{\mathrm{d}}{\mathrm{d}t}\left\{u(t) \otimes h(t)\right\}$$

- In the 2D case, the t here is one direction, you'd need two orthogonal lines to determine the full PSF

Inverse filtering

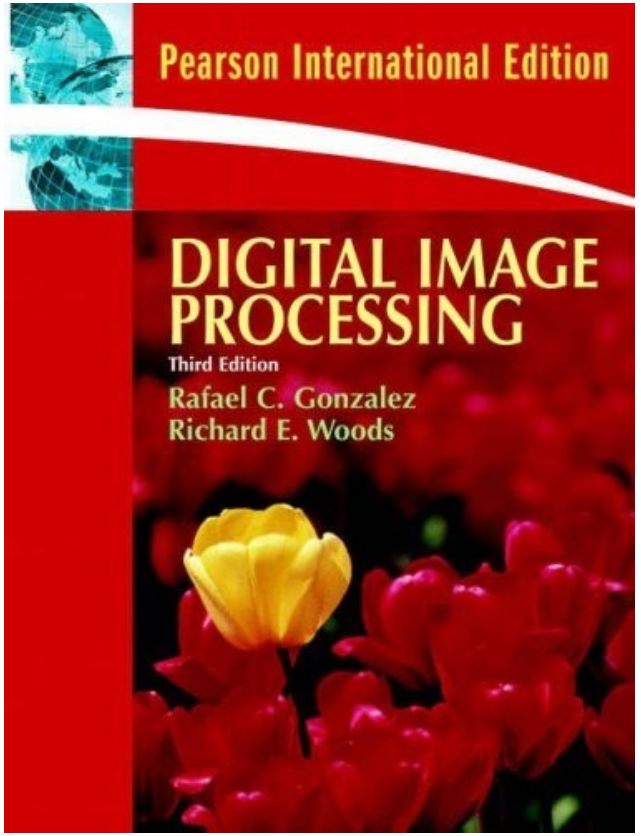
Image with motion blur
and more noise

Wiener filter,
after a good guess
of parameter K



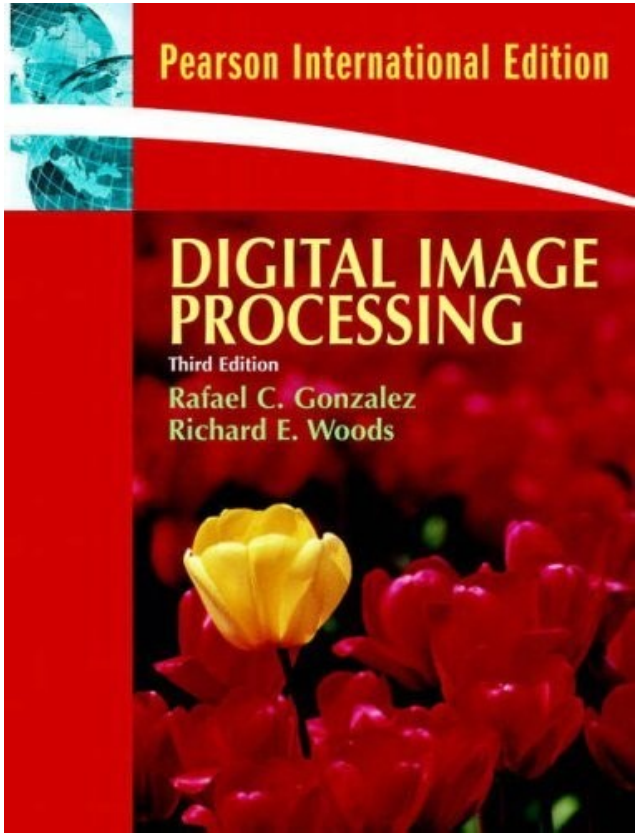
Reading assignment

- The Fourier transform and the DFT
  - Sections 4.2, 4.4, 4.5, 4.6, 4.11.1
- Sampling and aliasing
  - Sections 4.3, 4.5.4
- The FFT
  - Section 4.11.3
- Exercises:
  - 4.3
  - 4.4
  - 4.22



Reading assignment

- Convolution
  - Sections 4.2.5, 4.7
- Filtering in the Fourier domain
  - Sections 4.8, 4.9, 4.10, 5.4
- Deconvolution
  - Sections 5.5, 5.6, 5.7, 5.8, 5.9
- Exercises:
  - 4.21
  - 5.16
  - 5.18, 5.20



Summary of today's lecture

- All linear filters are convolutions
- Convolution in Spatial Domain is multiplication in Fourier domain:

$$f(x) \otimes h(x) \quad \underset{\Rightarrow}{\overset{\mathcal{F}}{}} \quad F(\omega) H(\omega)$$

- This allows designing linear filters in the Fourier domain
- Distortions such as lack of focus and motion blur can be corrected using Wiener filtering or more advanced deconvolution approaches