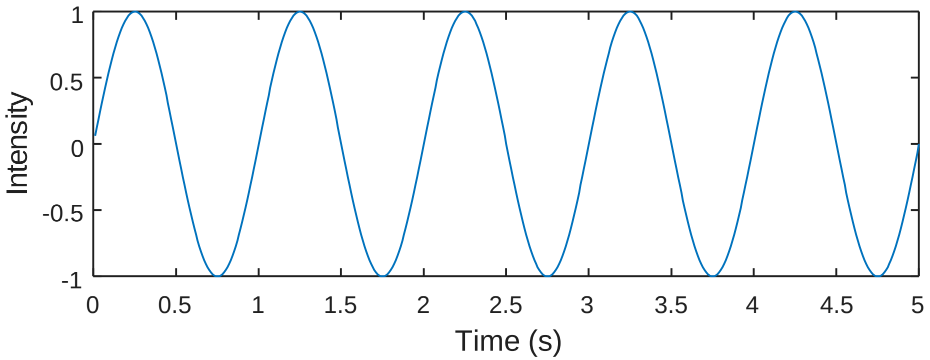
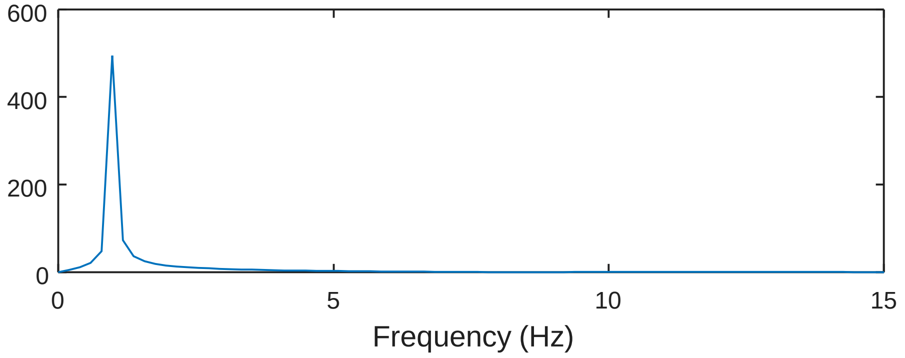
Fourier transforms, SIM

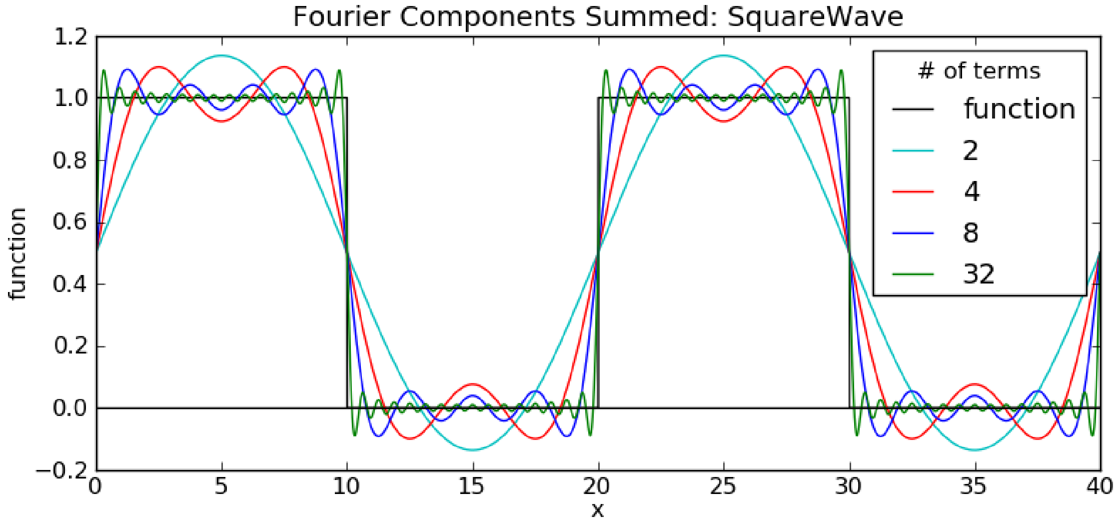
- Last class
  - More STED
  - Minflux
  - Fourier transforms
- This class
  - More FTs
  - 2D FTs
  - SIM

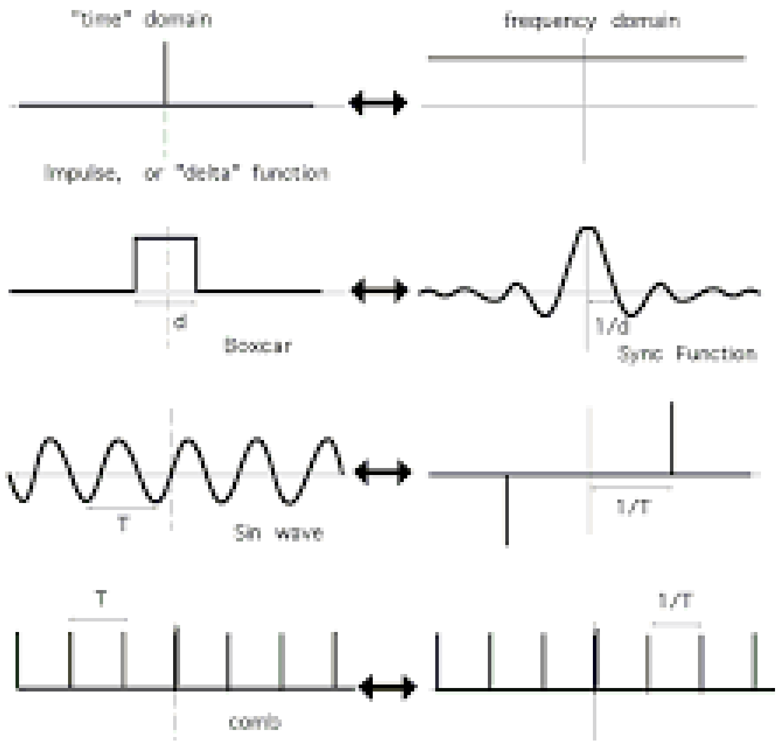
FT





IFT



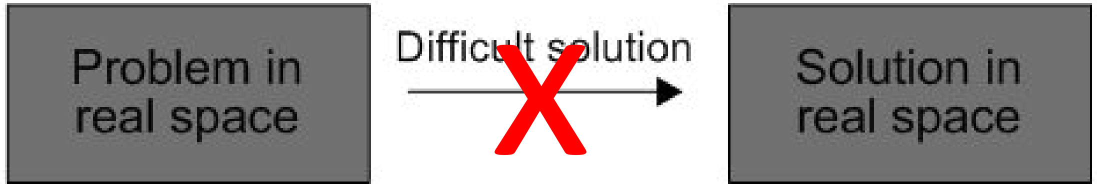


$$f(t) = \sum_n c_n \sin(2\pi n f)$$

If we can figure out c_n we can describe any function as a sum of sine waves

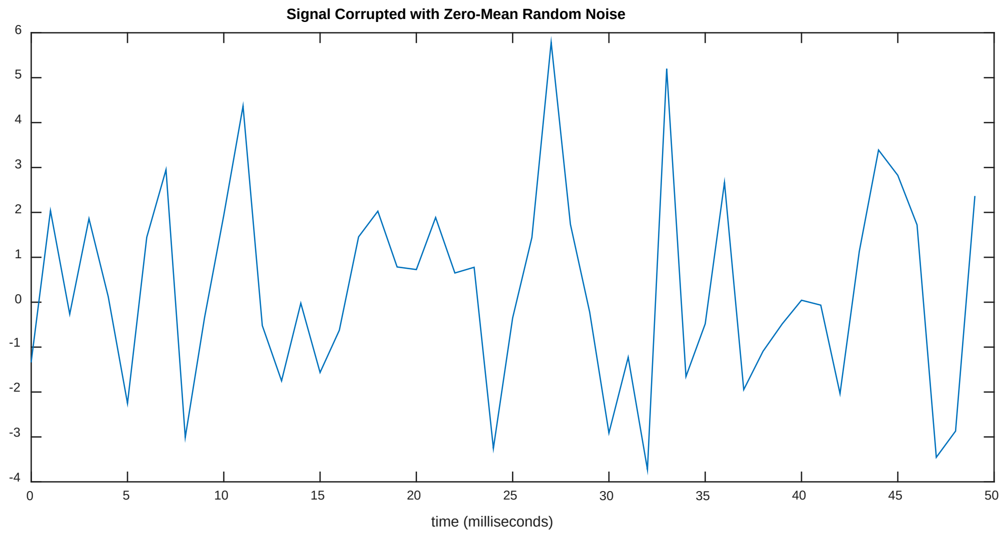
Why do we convert to frequency domain

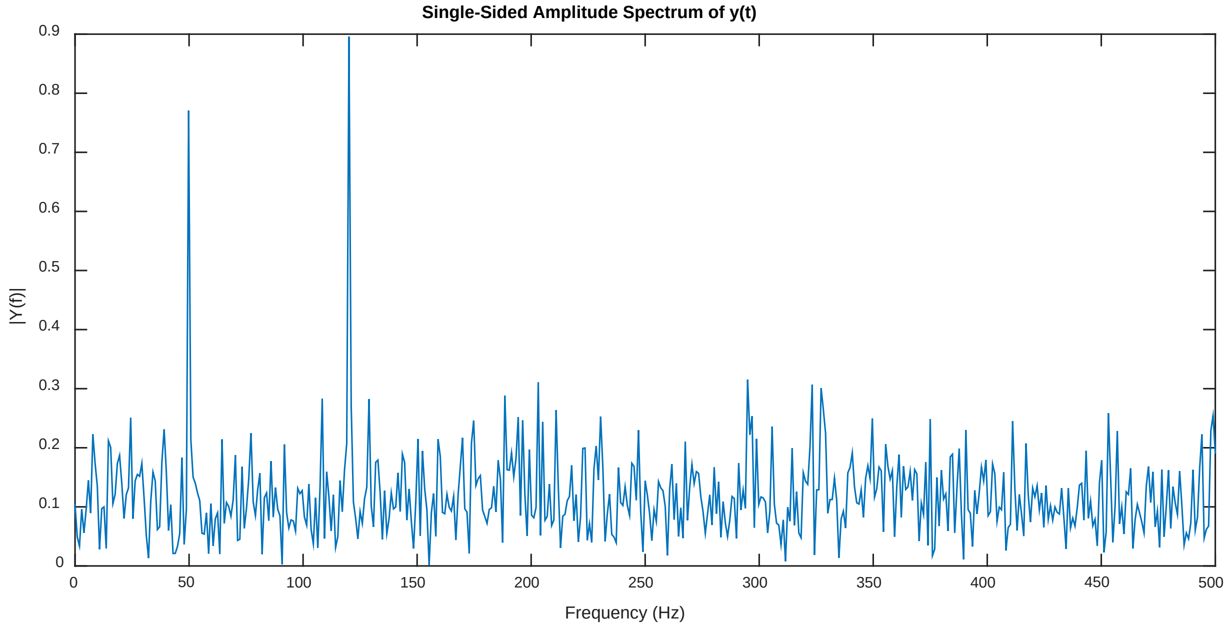
- Some problems are very hard to solve in the time domain
- Imagine a repeating source of noise
- There may be a very easy solution in the frequency domain
- It’s easy (in Matlab) to convert to frequency, apply the correction, and invert back to time domain



Fourier transforms useful to pick out signal in noise

- If you think you have an oscillating signal, but it is too noisy, you can often pick it out in frequency space
- Noise is often distributed randomly in frequency space, but your signal remains at one peak
- Conversion from time to frequency domain can clear things up
- Frequency space is heavily used in audio processing

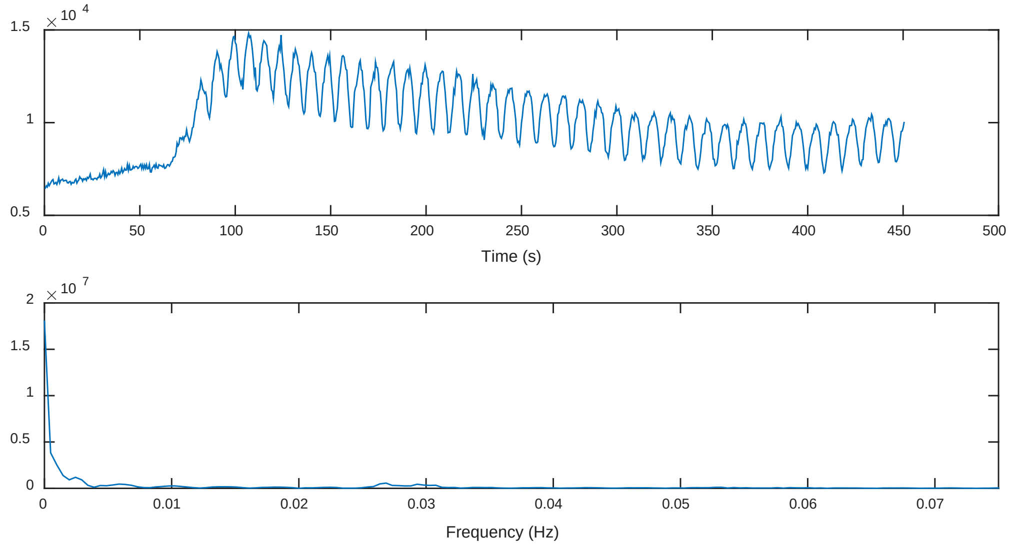




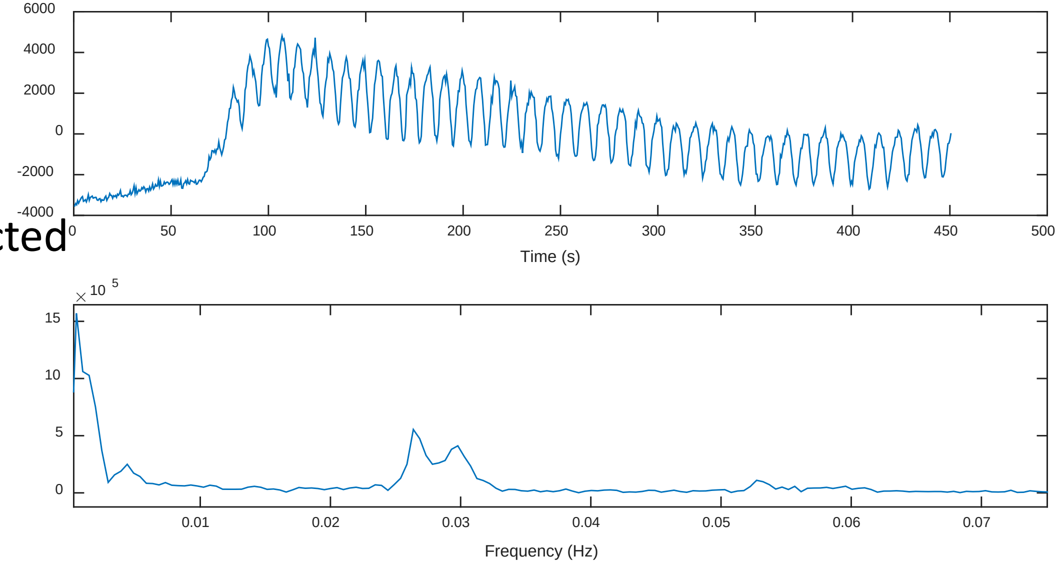
DC components of signals

- If the mean of your signal is ≠ 0, then you will have a DC component (freq = 0)
- This can overwhelm your lower frequencies
- Common to subtract the mean off your signal first
- Can also take the log of the frequencies to display nicely

Original

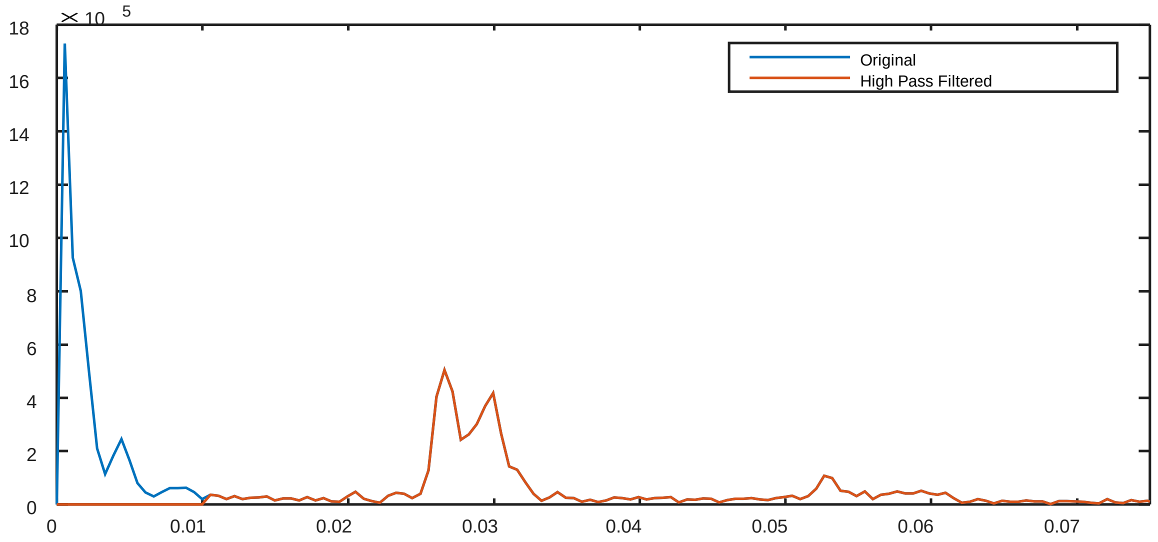


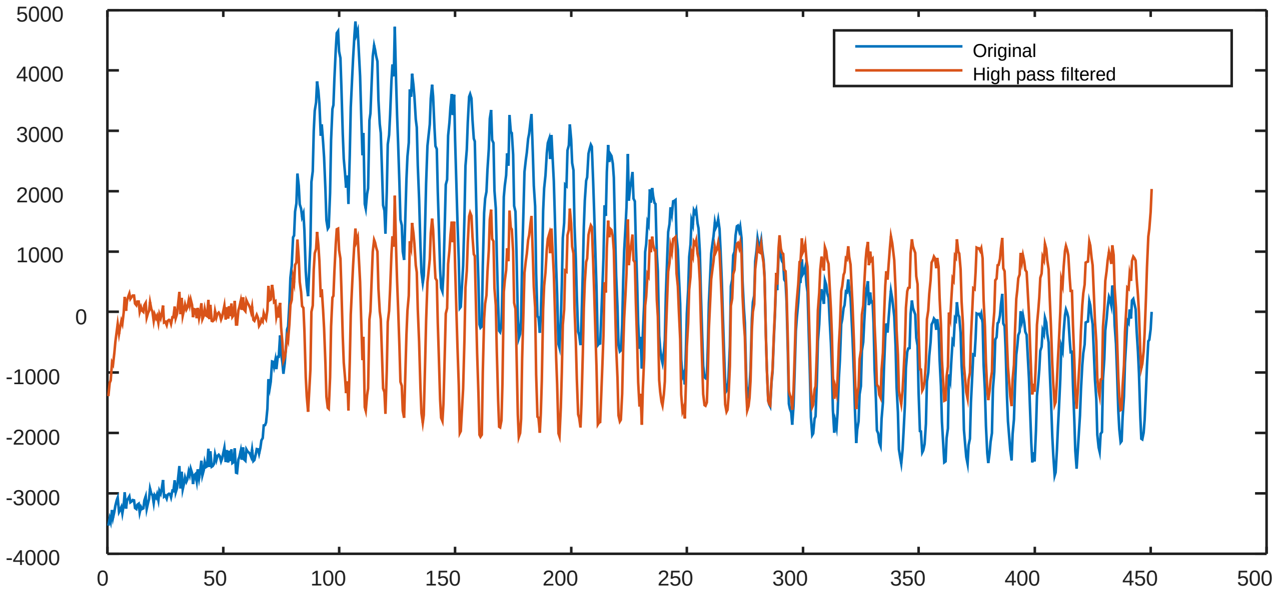
Mean subtracted



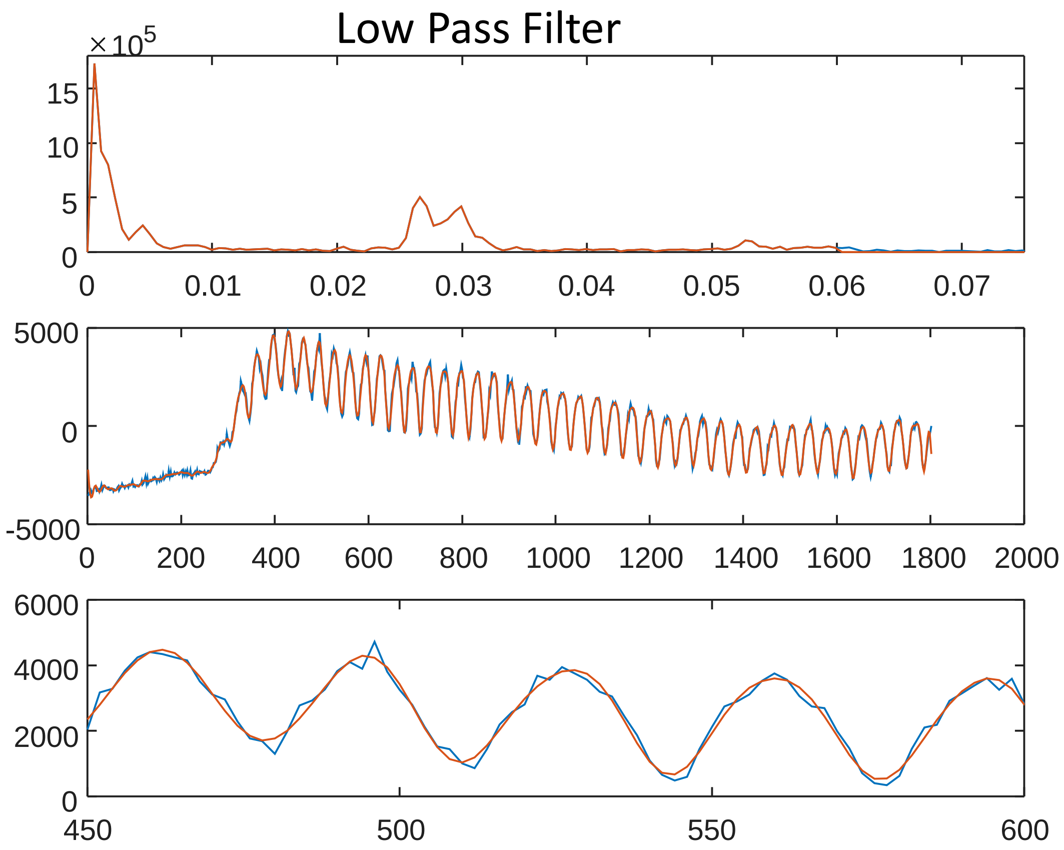
Fourier filtering

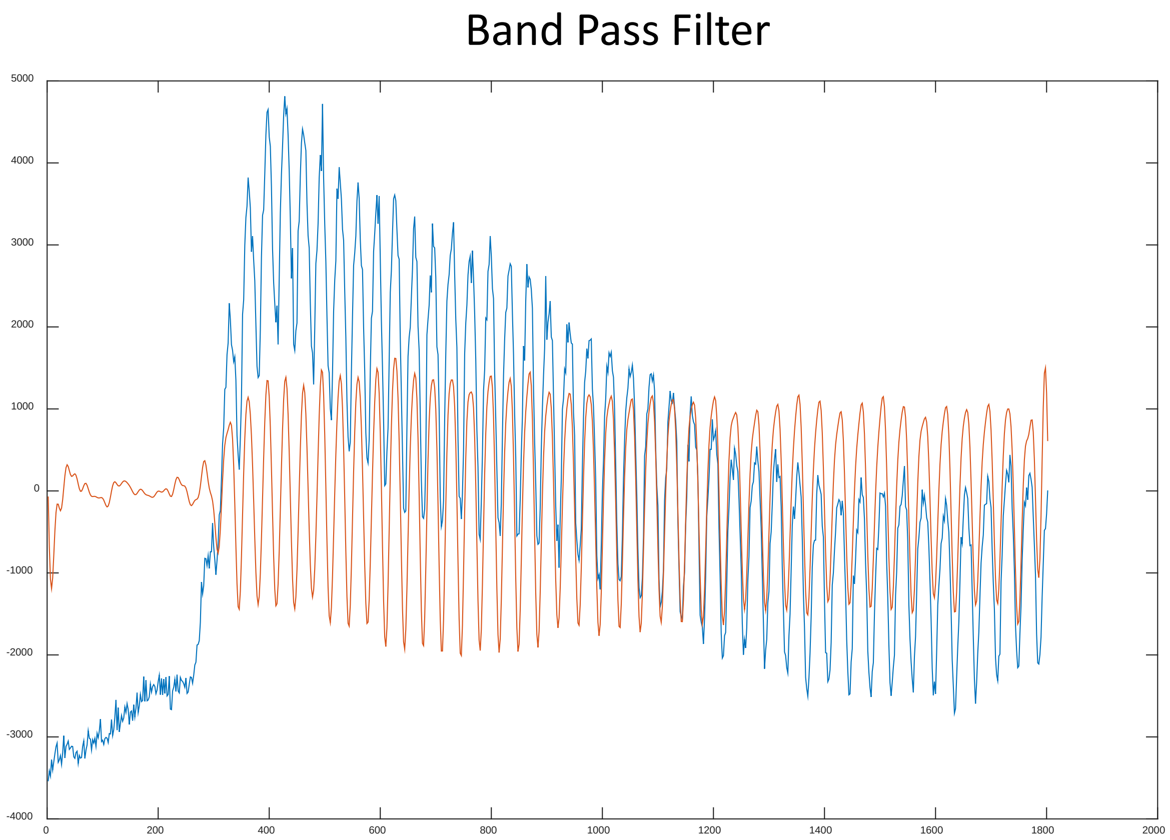
- If we know that the signal we care about is going to fall within some range
- It is easy to:
  - convert to frequency space
  - Set unwanted frequencies to 0
  - Inverse Fourier transform
- Very easy to set low pass, high pass, or band pass filters





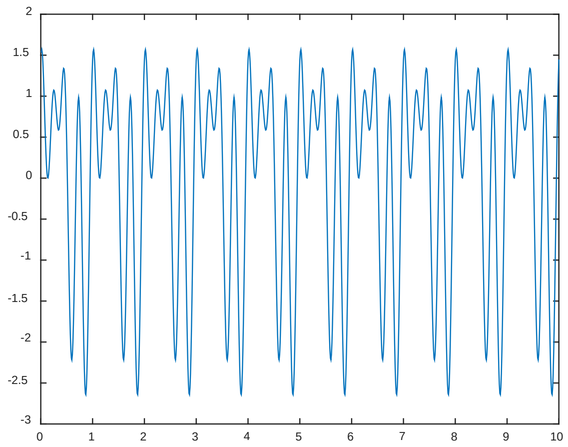
More Fourier filtering



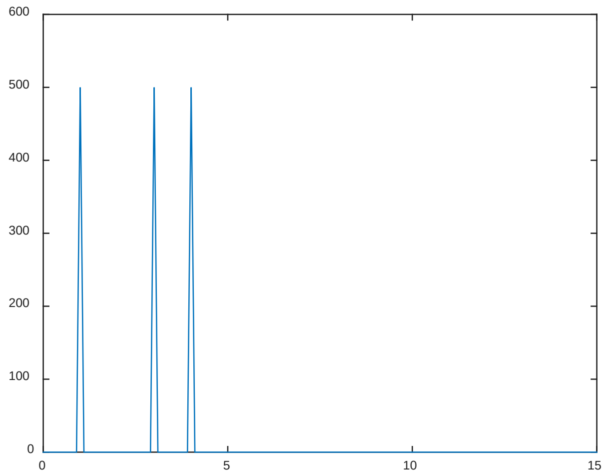


Fourier transforms encode phase as complex number

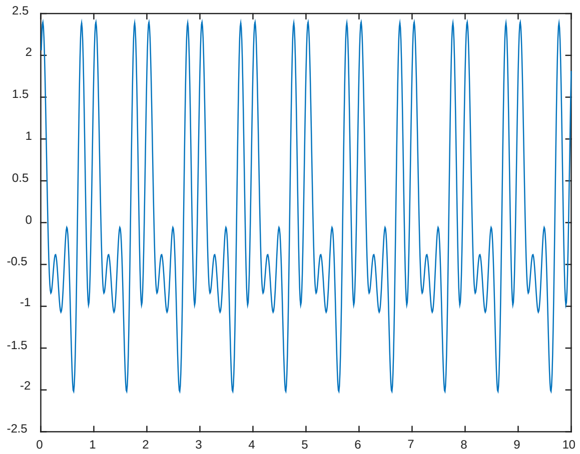
- Consider 3 sine waves with different frequencies and phases
- The FFT will look exactly the same independent of the phases
- In order to reconstruct the time varying signal, we need the phases




```
y1 = sin(2*pi*t + deg2rad(0));
y2 = sin(6*pi*t + deg2rad(30));
y3 = sin(8*pi*t + deg2rad(110));
```



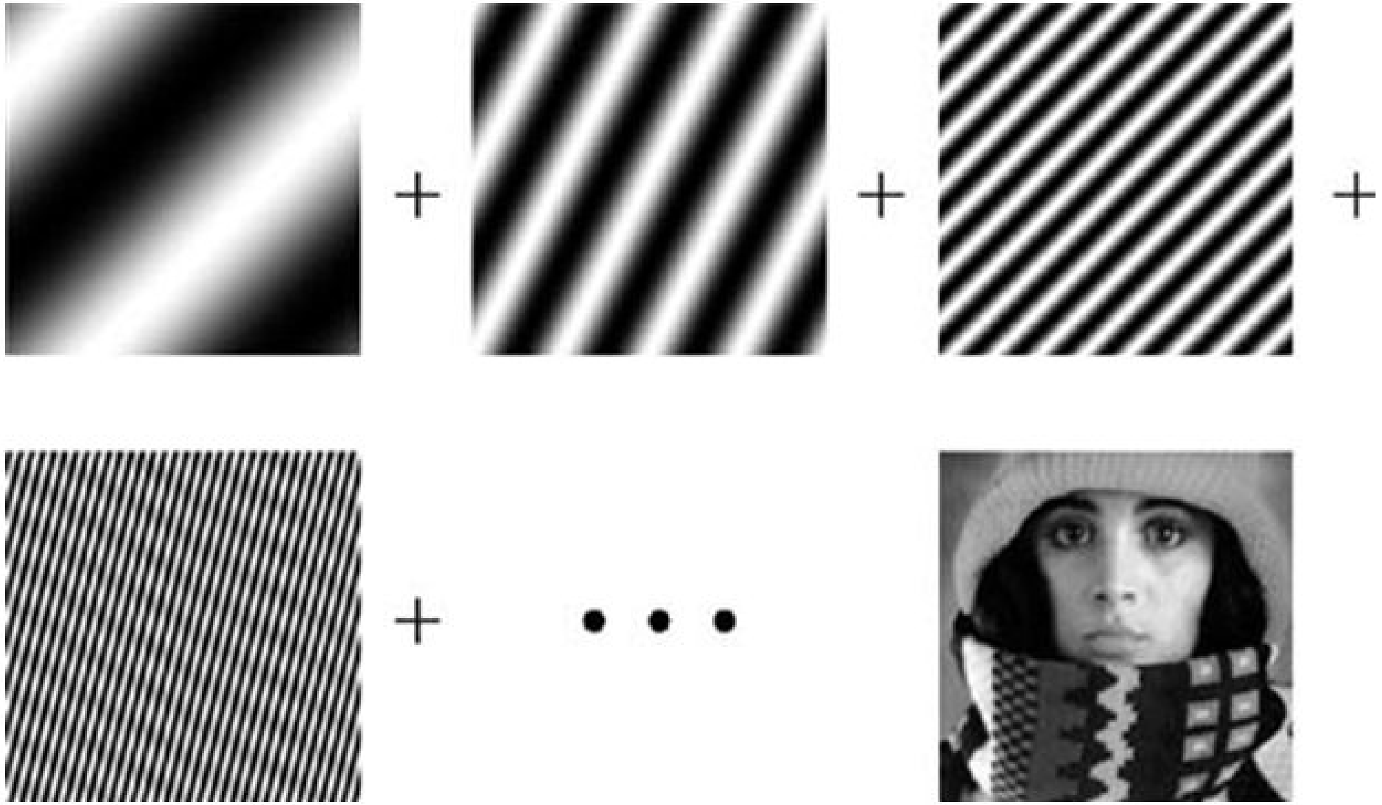
Note, I plotted the absolute value of the fourier transform



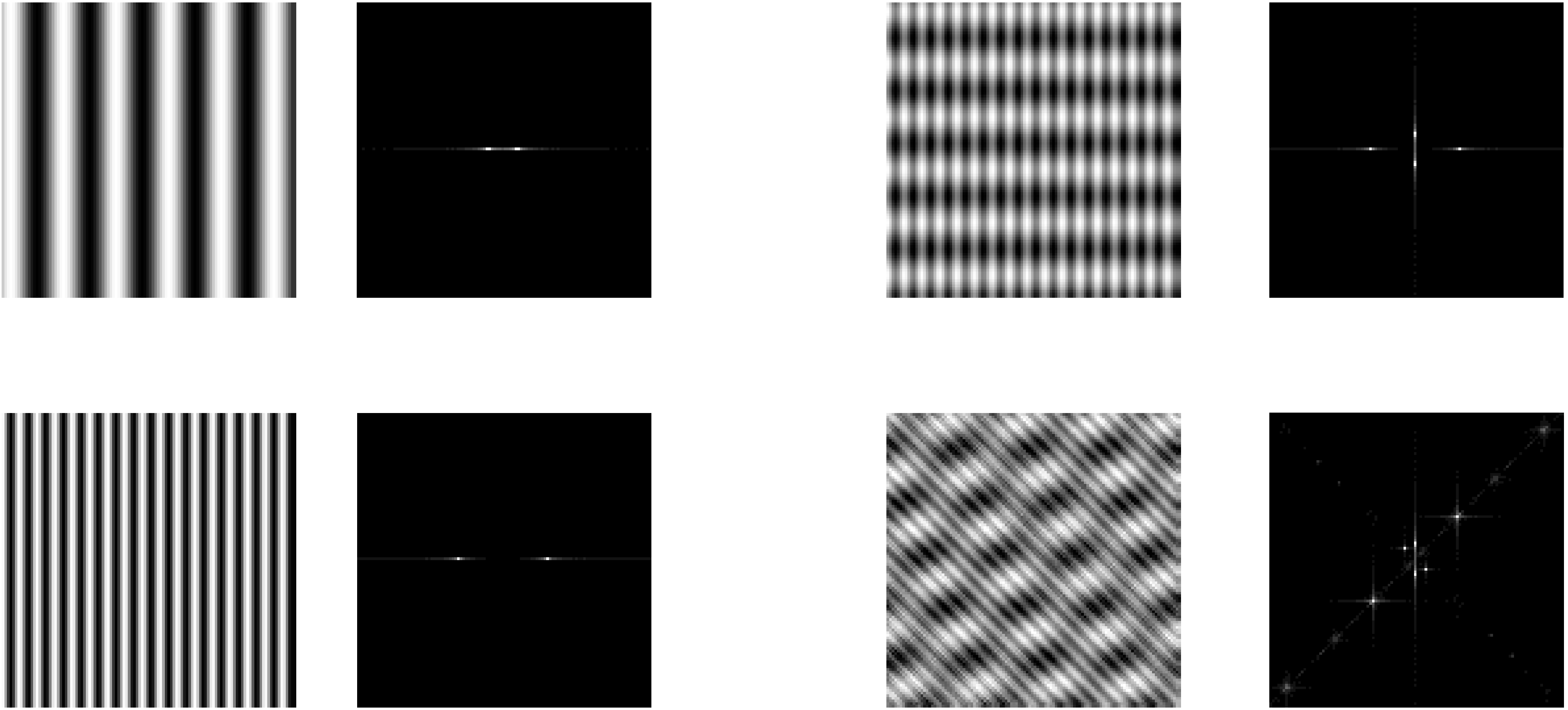

```
y4 = sin(2*pi*t + deg2rad(110));
y5 = sin(6*pi*t + deg2rad(0));
y6 = sin(8*pi*t + deg2rad(60));
```

2D Fourier transform

- In exactly an analogous way, linear combinations of 2d sine functions can be combined to form any image
- If you can calculate the amplitude and phase of each set, you can reconstruct any arbitrary image
- Matlab has a 2D FFT function that allows you to calculate amplitudes very quickly
- Allows image processing in the frequency domain

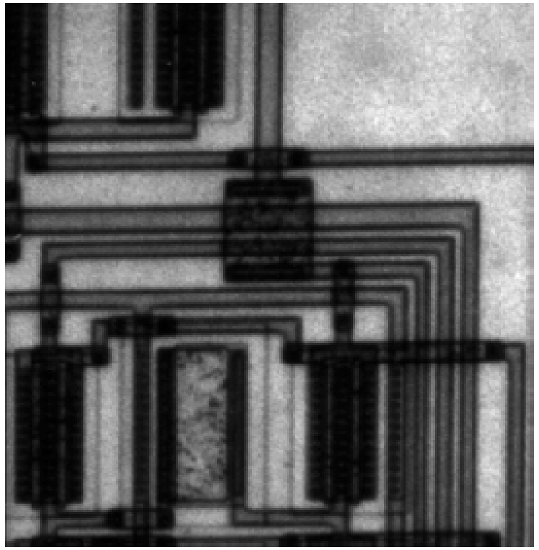
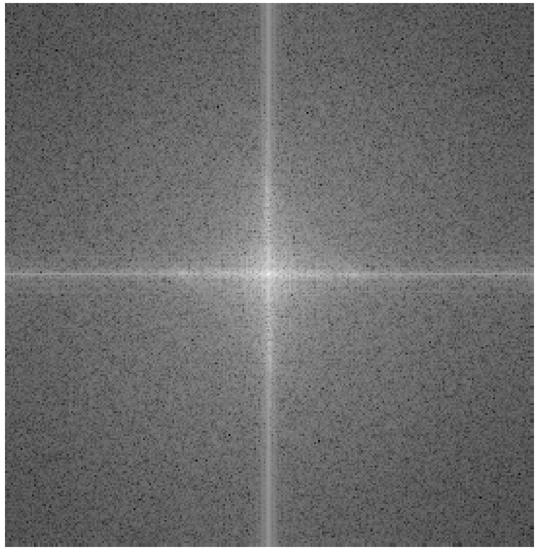
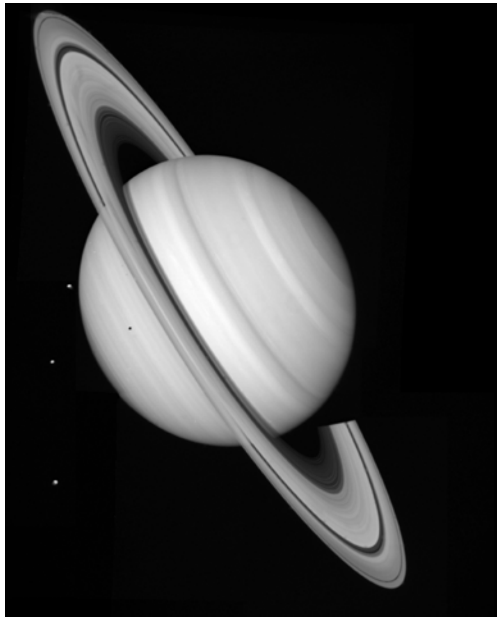
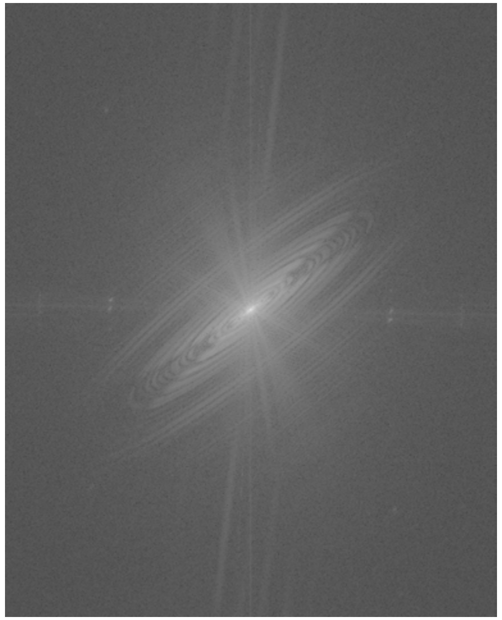


2D Fourier transforms



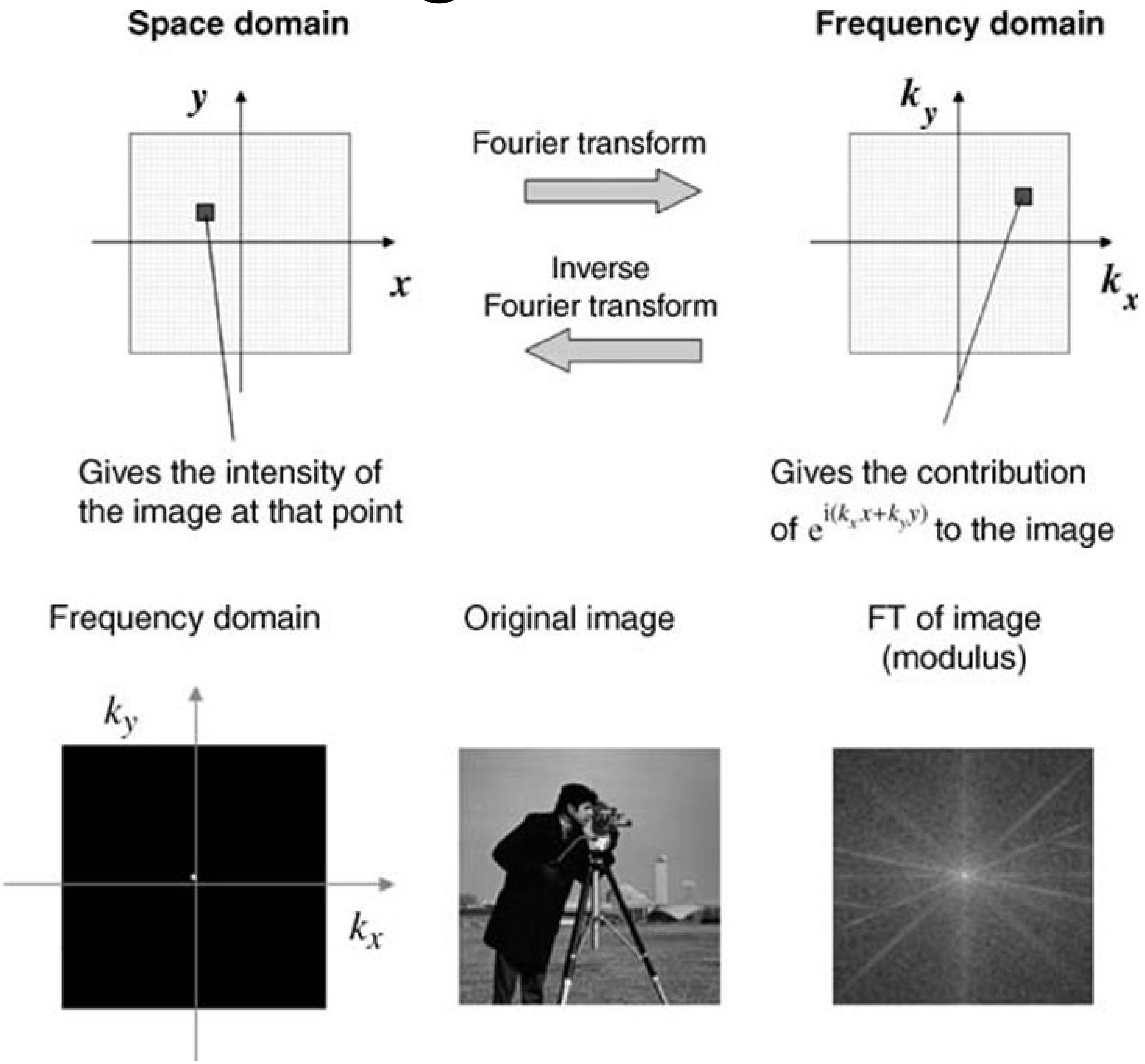
2D Fourier Transform of Images

- Now frequencies are represented by amplitude in space
- X axis (frequency) is periodic signal in y dimension, and vice versa
- The center of the image is the DC signal
- Diagonal points represent diagonal periodicity
- Edges represent the highest frequencies encoded

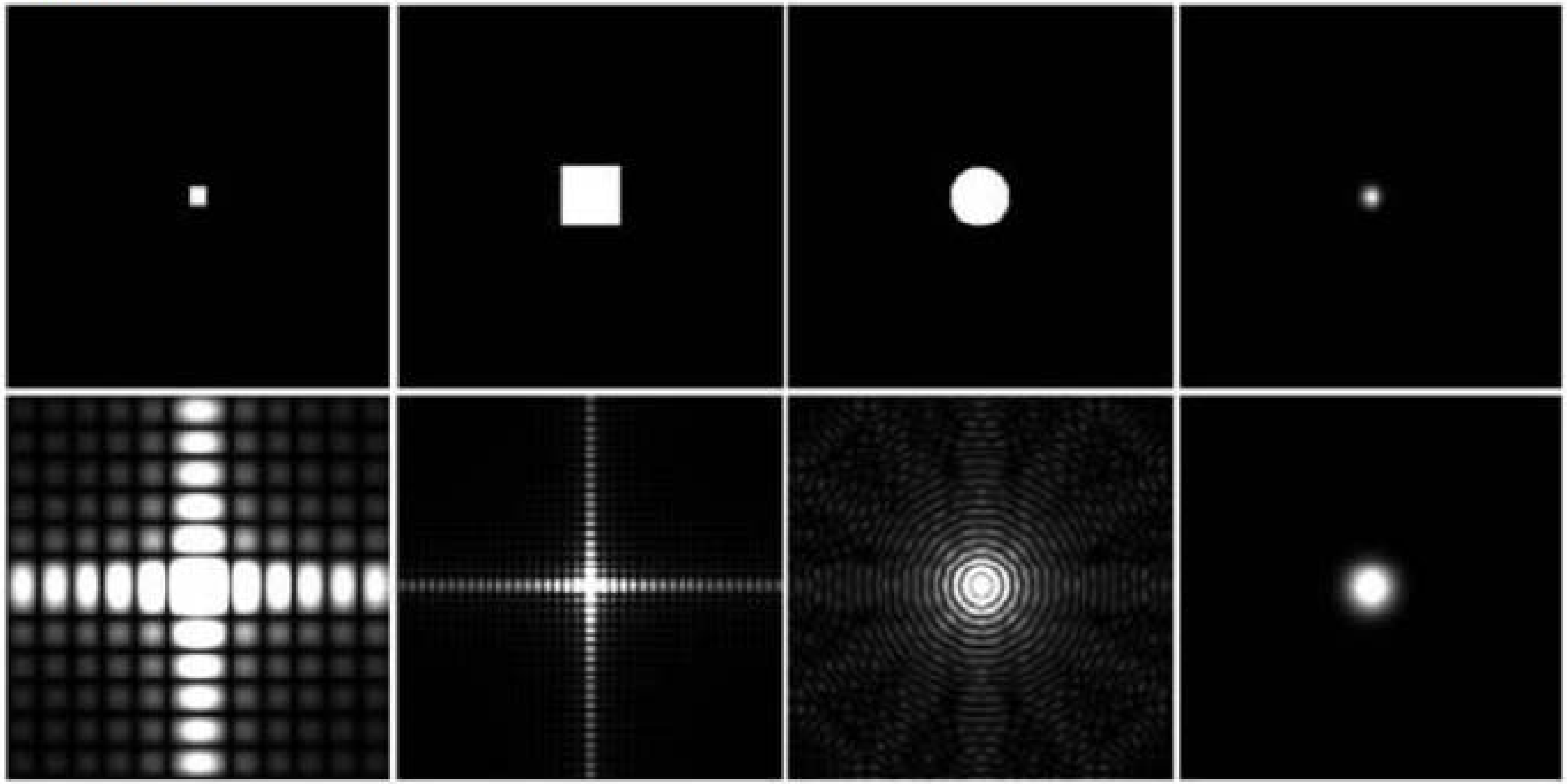
2D Fourier transform meaning

- In space, each pixel represents an intensity
- In frequency, each pixel represents the amount of that spatial frequency
- In the camera man, there are vertical lines (buildings), horizontal lines (skyline), and diagonal lines (tripod)

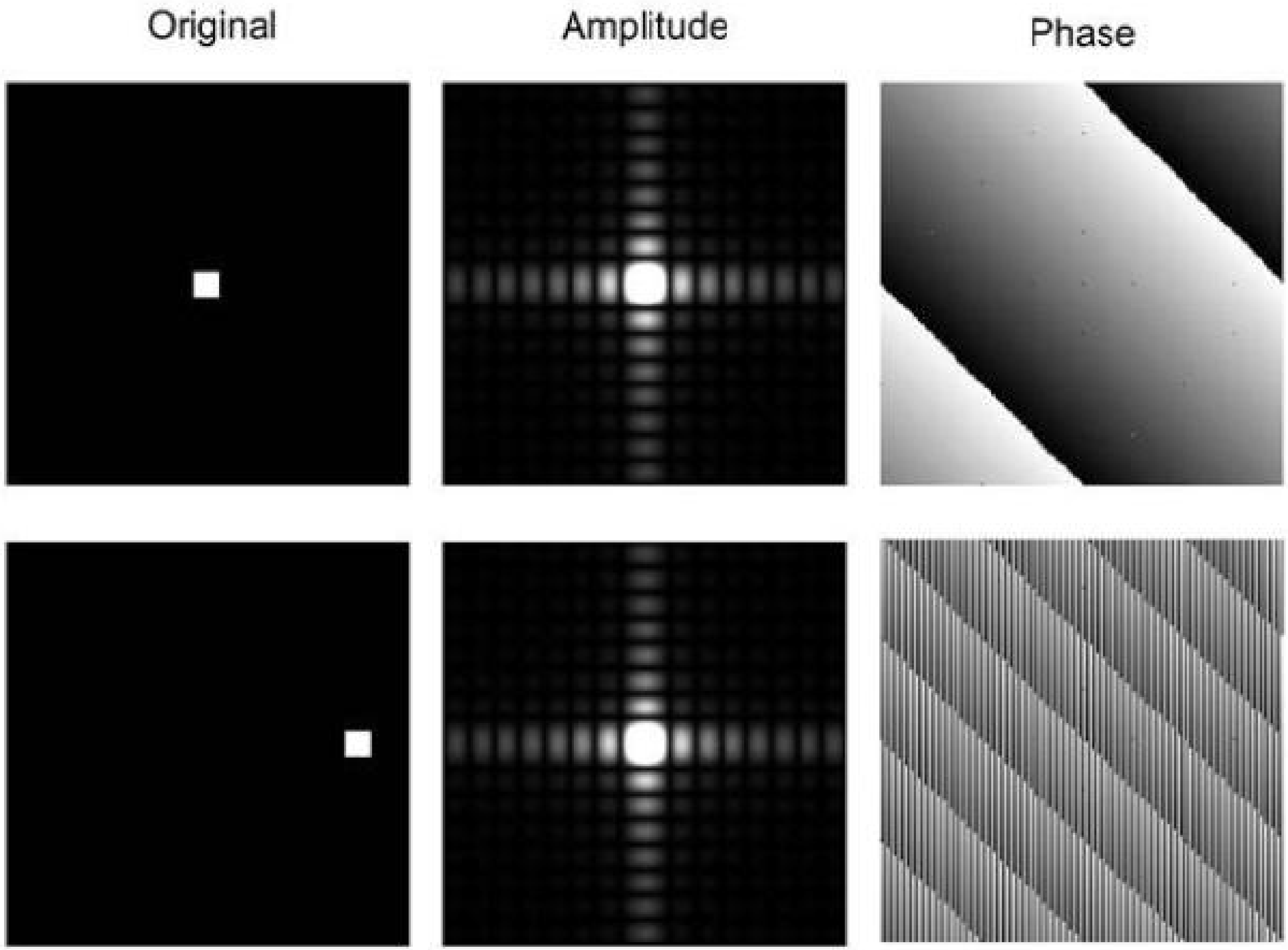


Amplitude spectra

Images

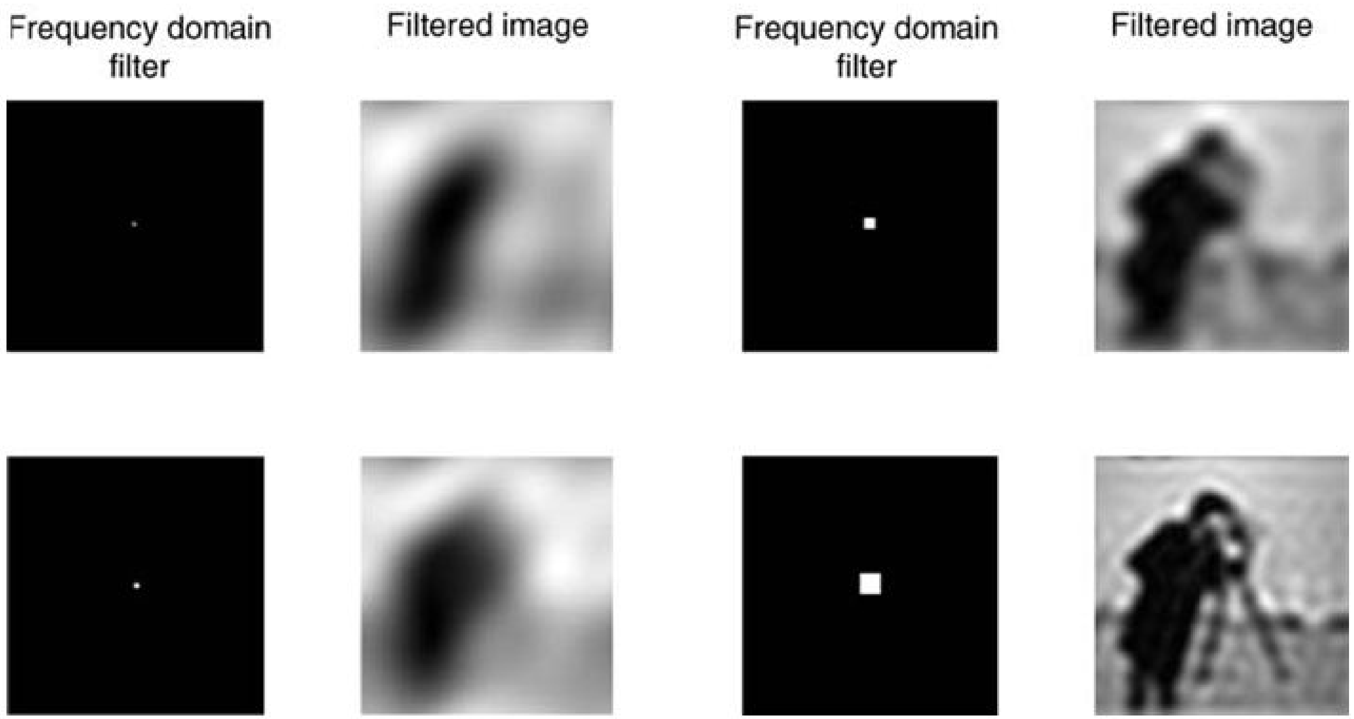


Fourier Transforms


Original
Amplitude
Phase

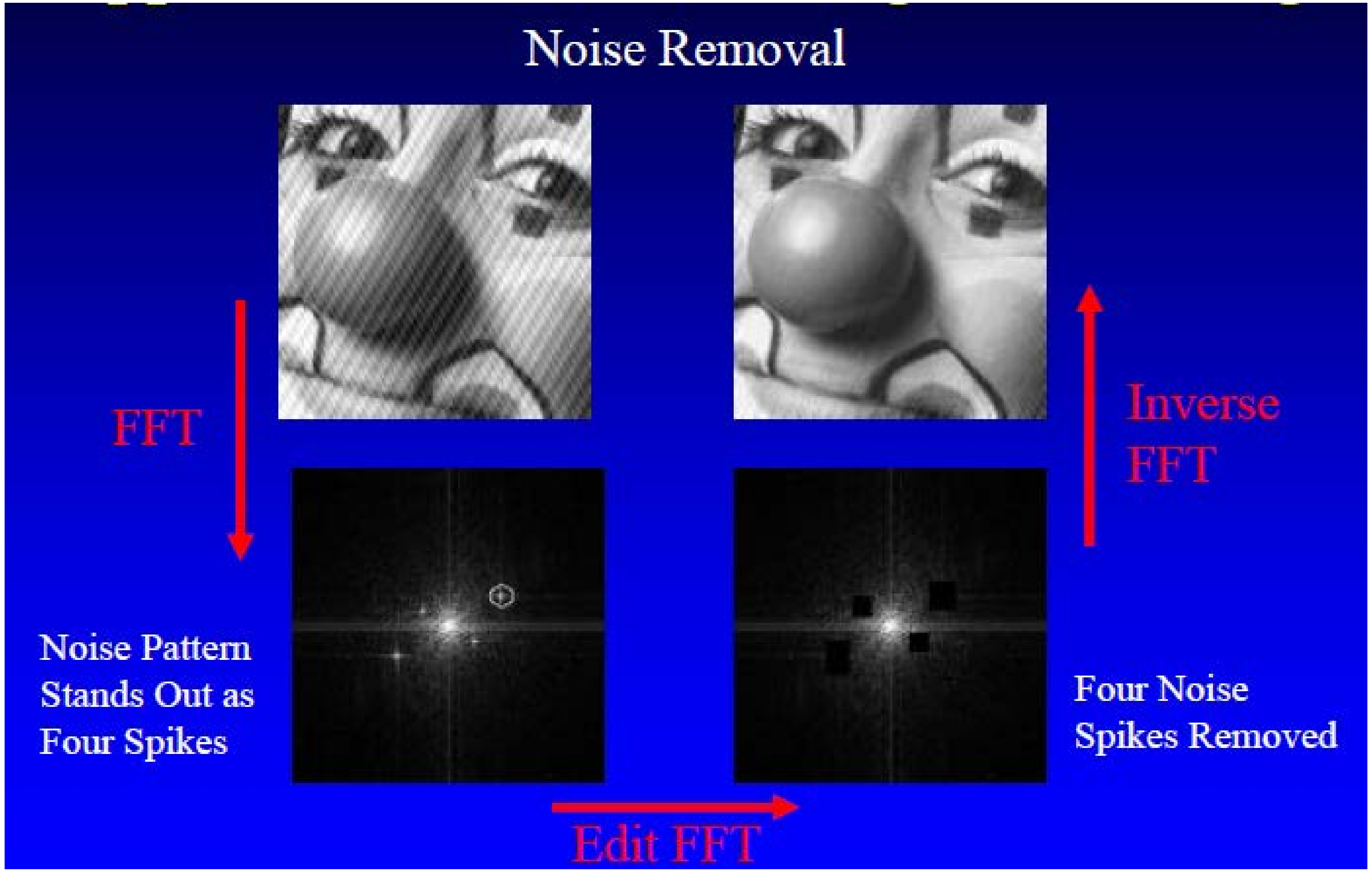
FFT and filtering

- The more frequencies encoded in an image, the sharper the detail you can resolve
- By restricting higher frequencies you can filter the image



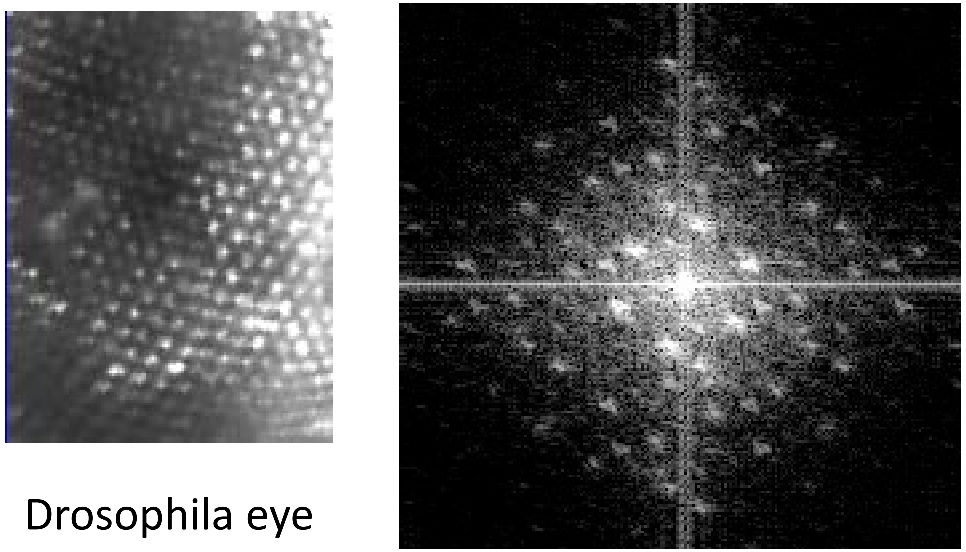
Applications: Noise removal in images

- Similar to the 1D case, periodic noise can be easily suppressed
- Regular noise appears as points
- Set those points in frequency space = 0
- IFFT to convert back to spatial coordinates

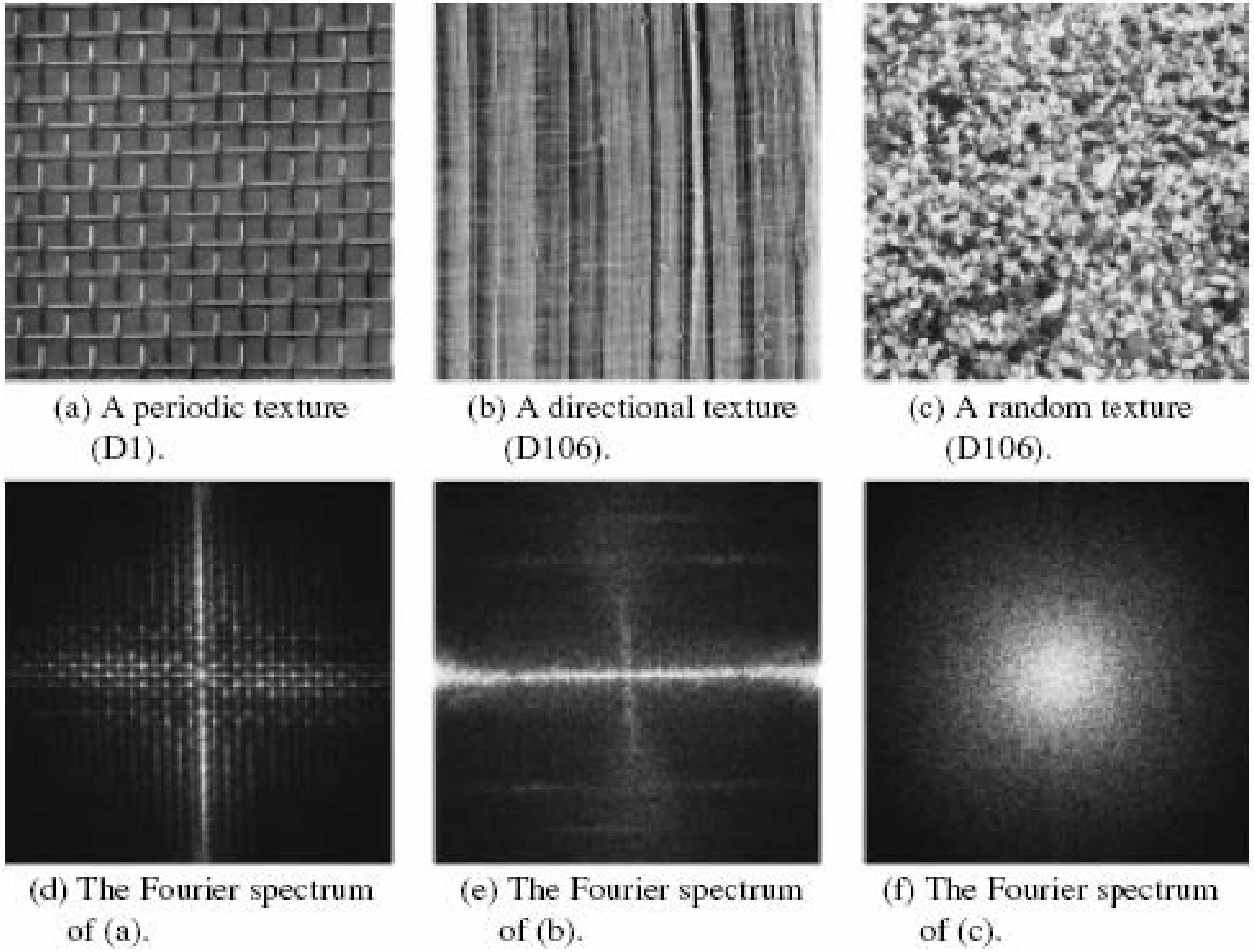


Applications: recognition of textures

- Repeating textures will have distinct Fourier components
- Easy to pick them out in frequency space



Drosophila eye and it's 2D FFT



(a) A periodic texture (D1).

(b) A directional texture (D106).

(c) A random texture (D106).

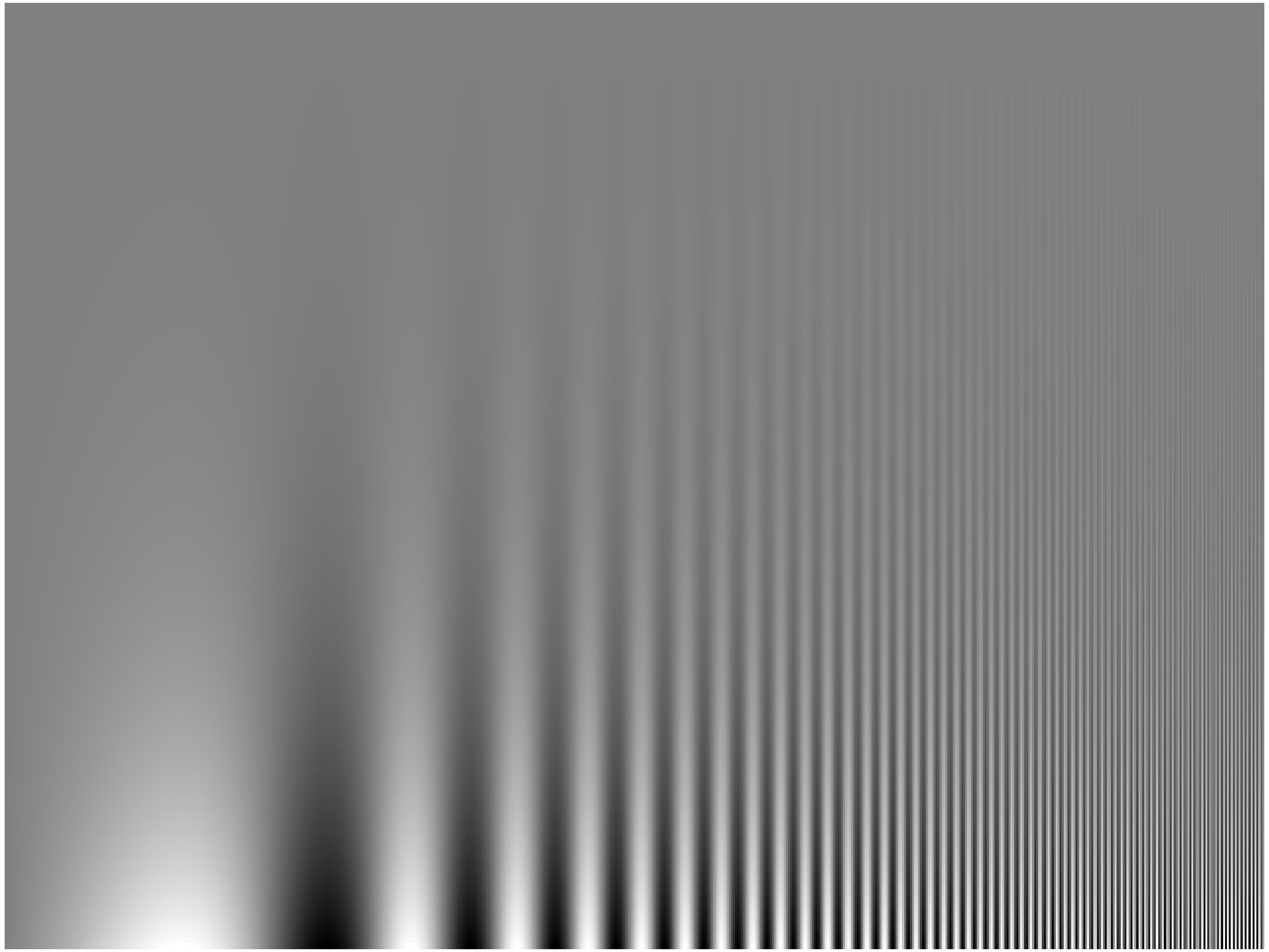
(d) The Fourier spectrum of (a).

(e) The Fourier spectrum of (b).

(f) The Fourier spectrum of (c).

Frequency sensitivity of visual system

- Campbell Robson sensitivity curve
- The U shape is a pattern of your visual system, not the image
- The computer doesn't care, but you (and readers) can misinterpret images that contain too high or low of frequencies



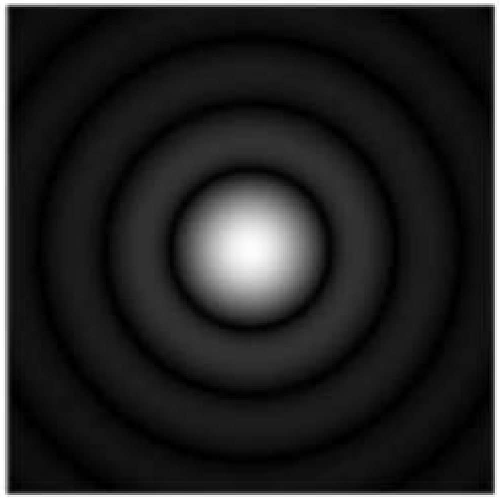
SIM

- 3rd class of superresolution

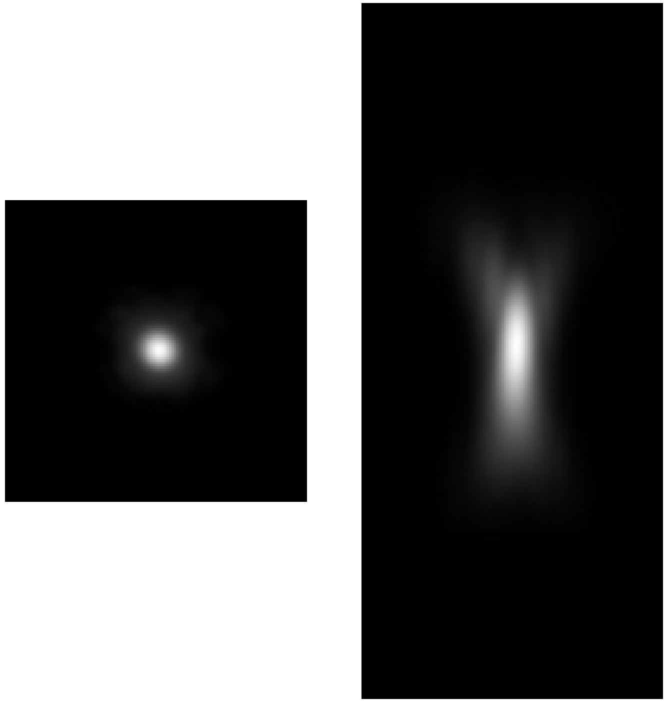
Origins of diffraction limit

- The airy disk size is determined by the wavelength and NA
- The resolution between two objects is set by Rayleigh criterion
- It is the maximum angle that sets our size

Airy Disk

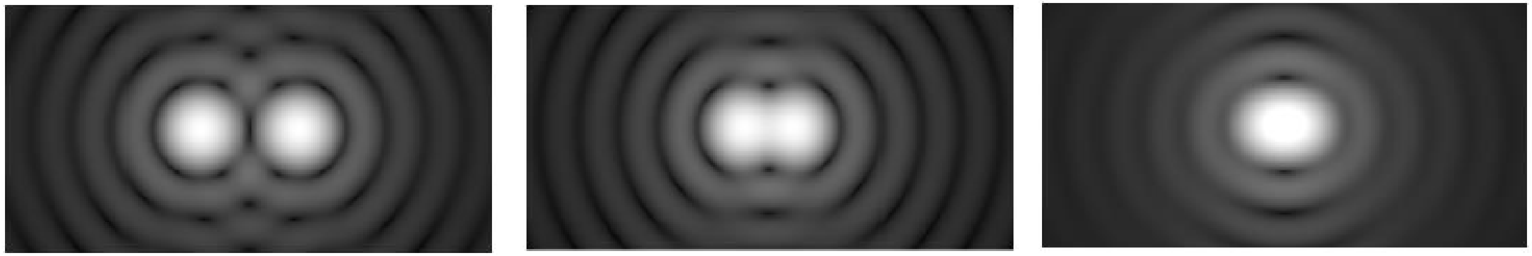


Point Spread Function



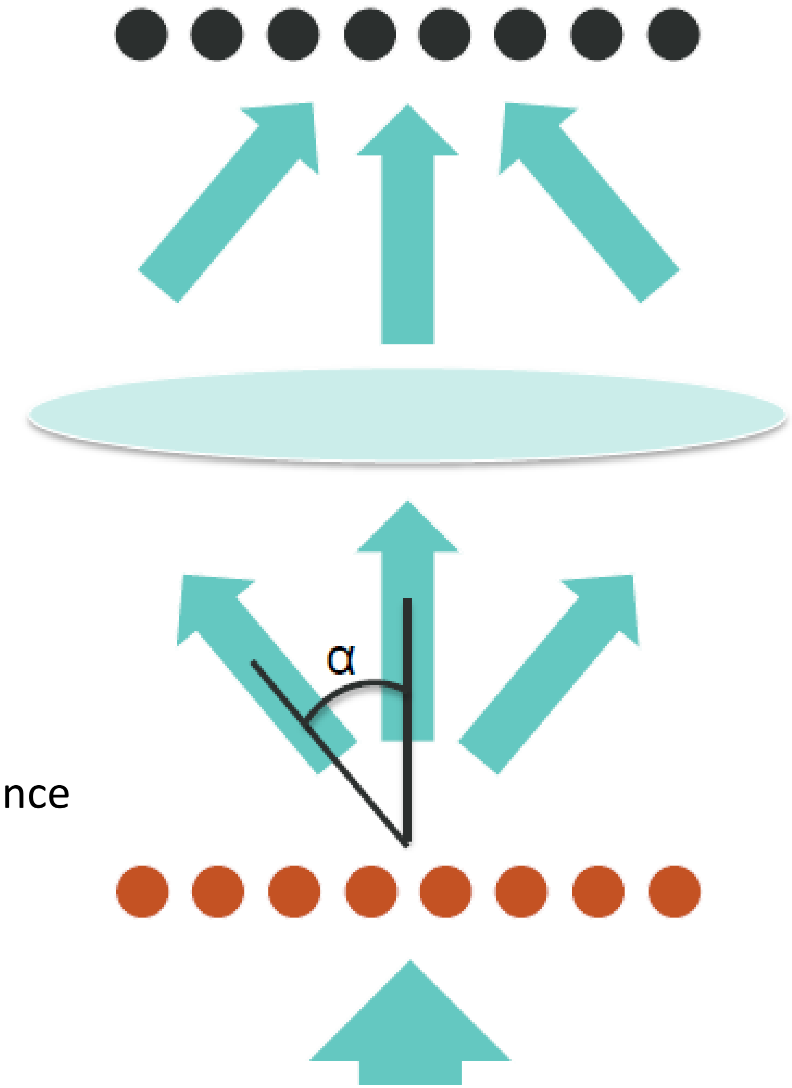
$$R = 1.22\frac{\lambda}{2\mathrm{nsin}(\theta)} = .61\frac{\lambda}{NA}$$

Smallest distance at which we can resolve 2 points



Limit is result of sample diffracting light

- Abbe realized we can think of the sample diffracting the light
- The lens re-images those diffraction patterns back onto the sample plane
- The angle of diffraction is proportional to the spacing

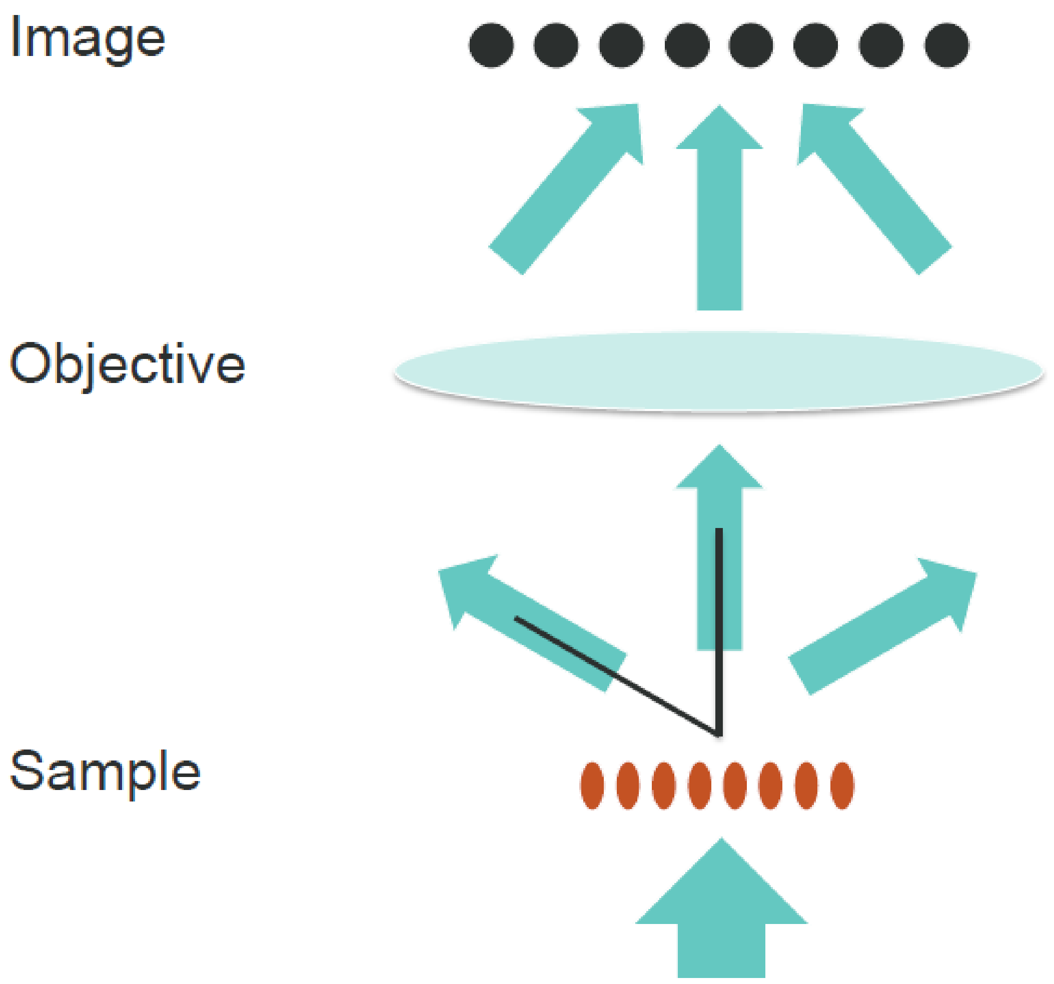


$$\sin(\theta) = \frac{\lambda}{2d}$$

d = sample structural distance

Effects of finite objective size

- The fact that we can't capture all the diffracted rays means that we lose the largest angles
- These angles correspond to the smallest features in the sample
- To resolve smaller features, we need to capture these higher frequencies



Optical transfer function

k = spatial frequency
$k_0 = 2NA/\lambda_{em}$ = maximum observable spatial frequency

- Optical transfer function is the fourier transform of the PSF
- Frequency space representation of how the image is formed on the camera
- Convolution in image space = multiplication in frequency space

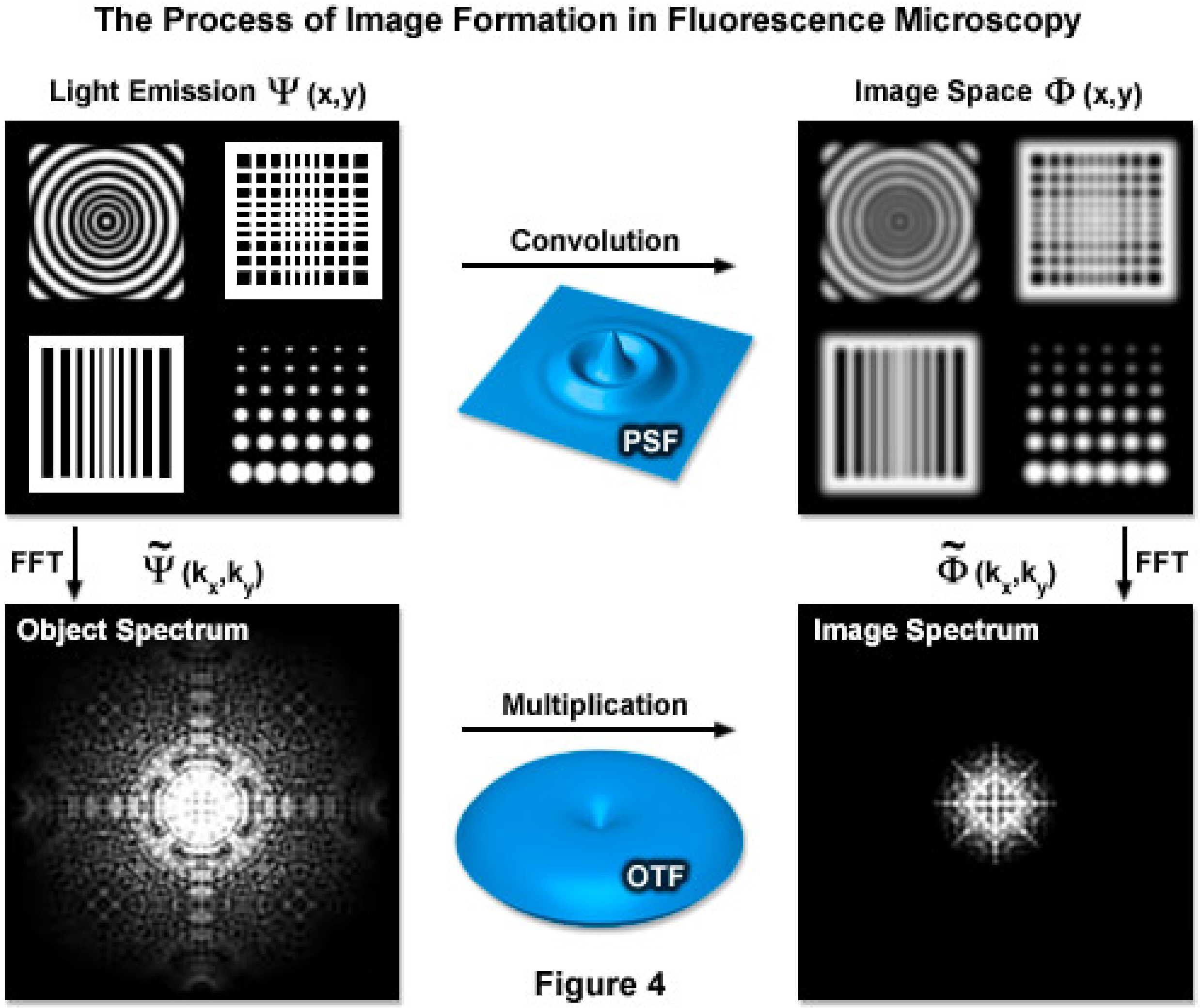


Figure 4

On to Matlab…

Applications: Help with anti-aliasing

- Aliasing is a feature of sampling two slowly
- Adds frequencies not present in original signal
- Shows up in images as blotchy regions