

>>> IMAGE PROCESSING AND  
COMPUTATIONAL PHOTOGRAPHY  
SESSION 3: FREQUENCY DOMAIN

Oriol Pujol & Simone Balocco

# REVIEW QUESTIONS

3. Fill in the blanks:

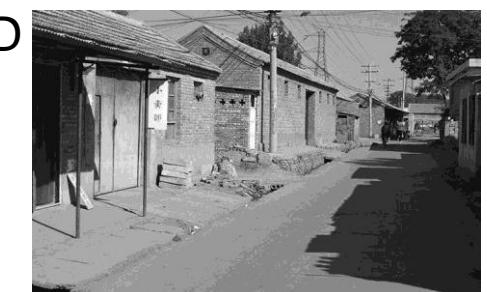
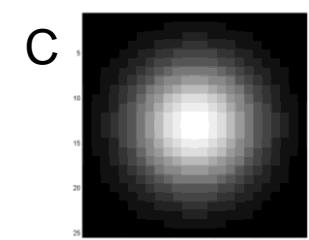
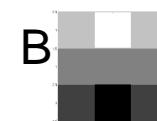
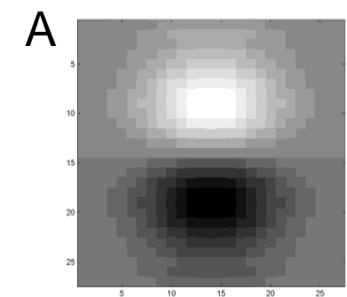
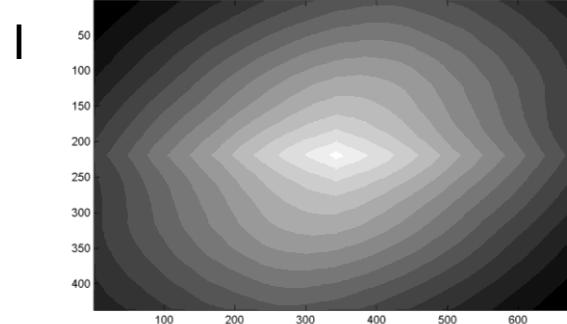
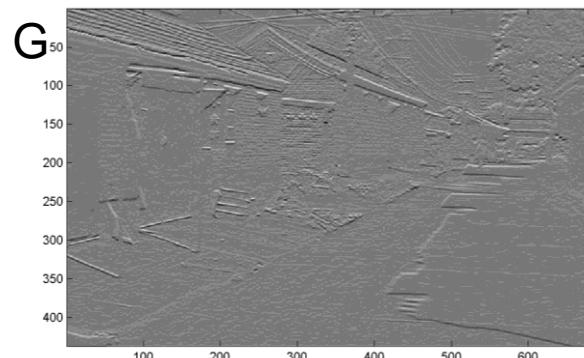
↓ Filtering Operator

a)  $\underline{\quad} = D^* B$

b)  $A = \underline{\quad}^*$

c)  $F = D^* \underline{\quad}$

d)  $\underline{\quad} = D^* D$

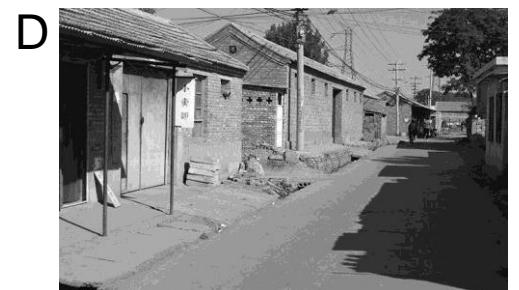
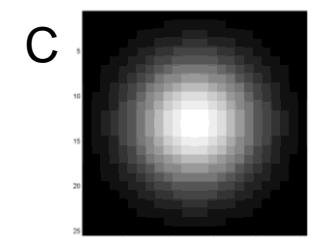
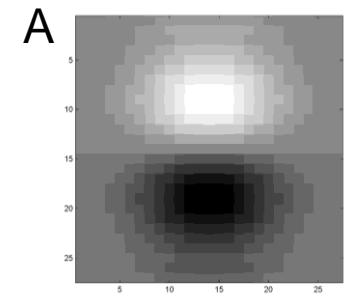
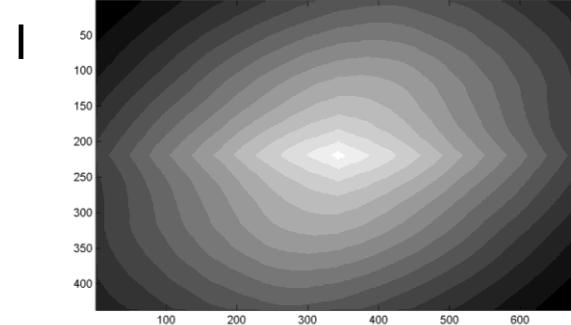
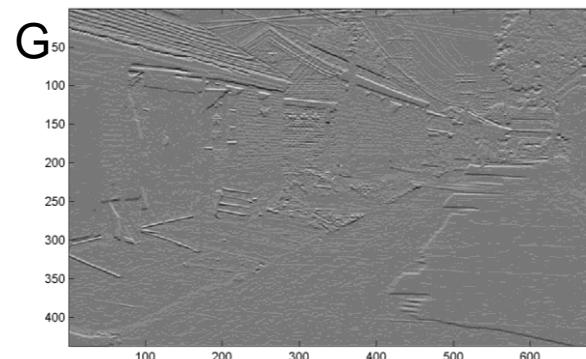


# REVIEW QUESTIONS

3. Fill in the blanks:

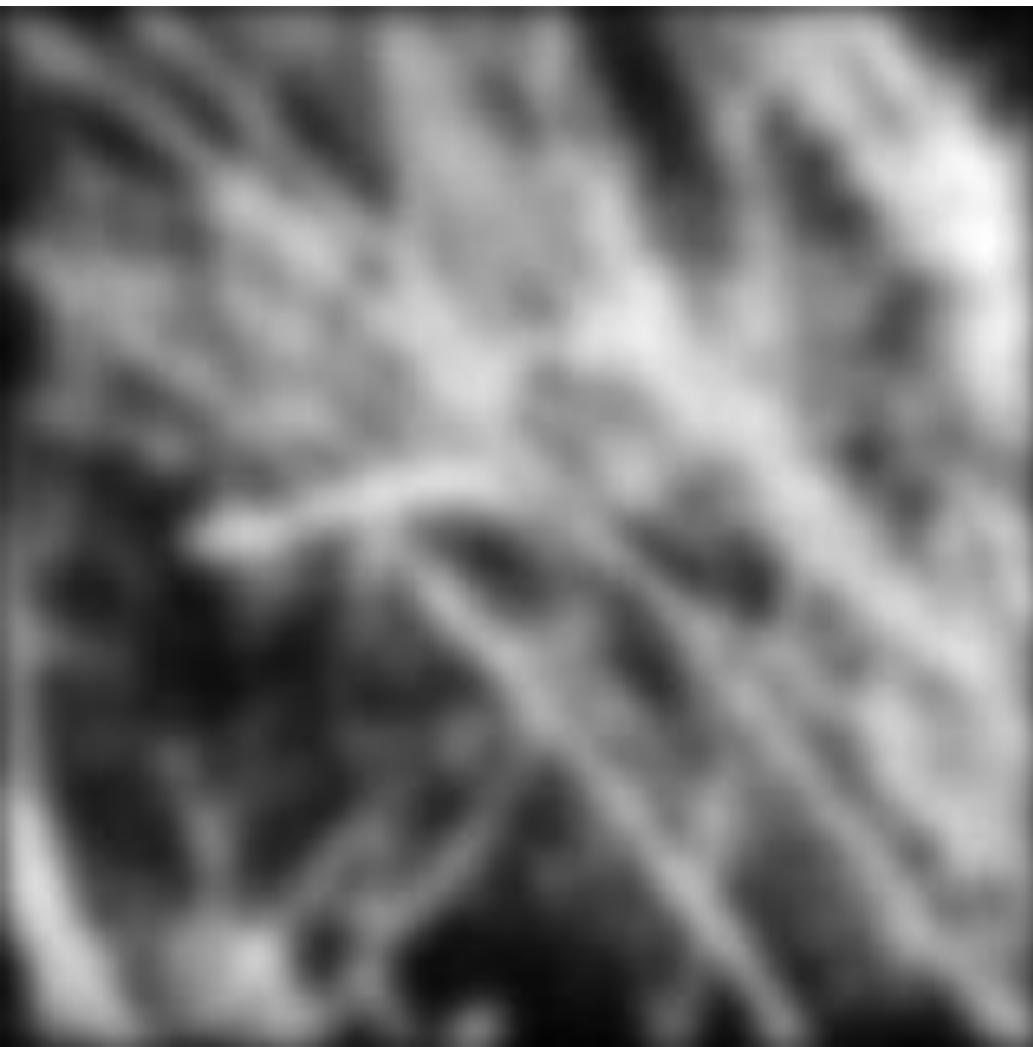
↓ Filtering Operator

- a)  $G = D * B$
- b)  $A = C * B$
- c)  $F = D * E$
- d)  $I = D * D$

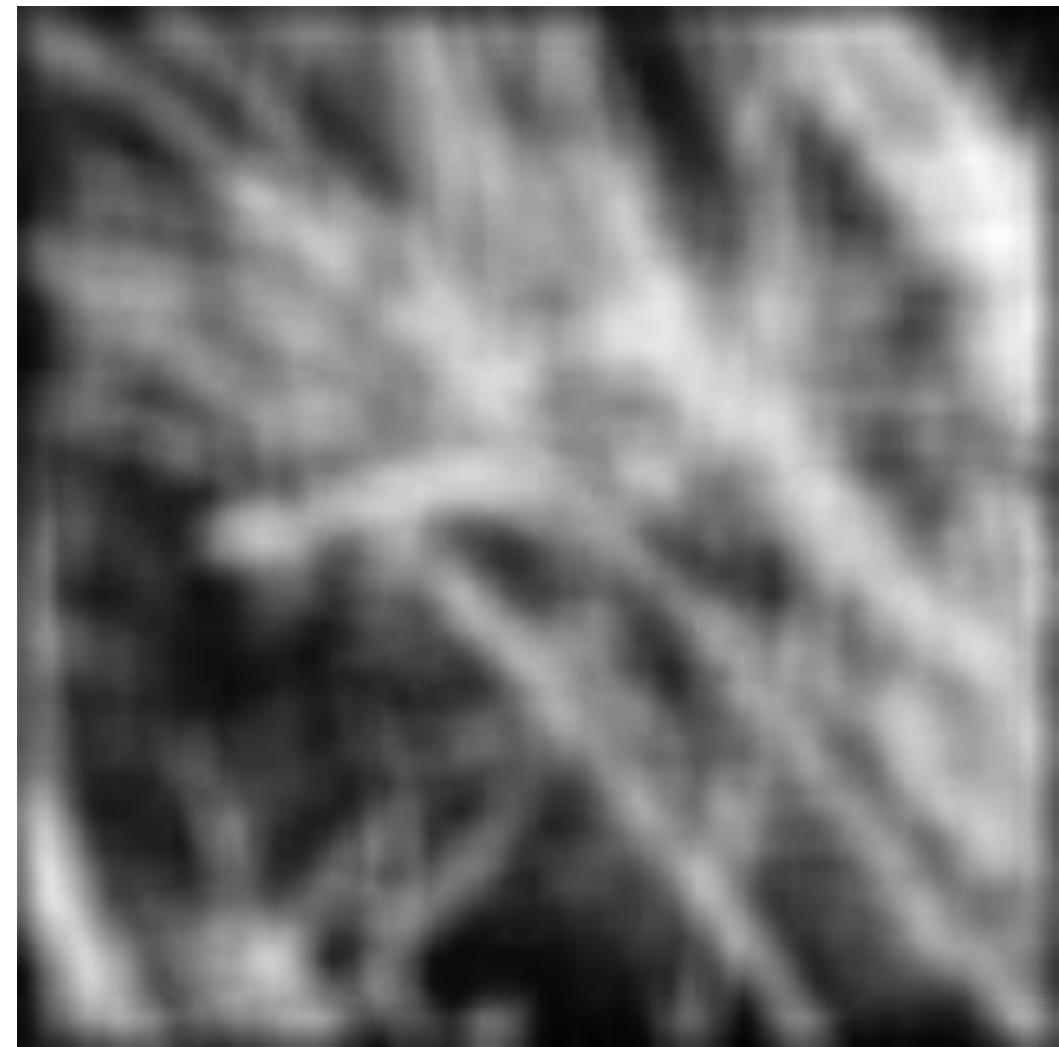


# WHY DOES BOX FILTER SMOOTHING HAVE SO MANY ARTIFACTS?

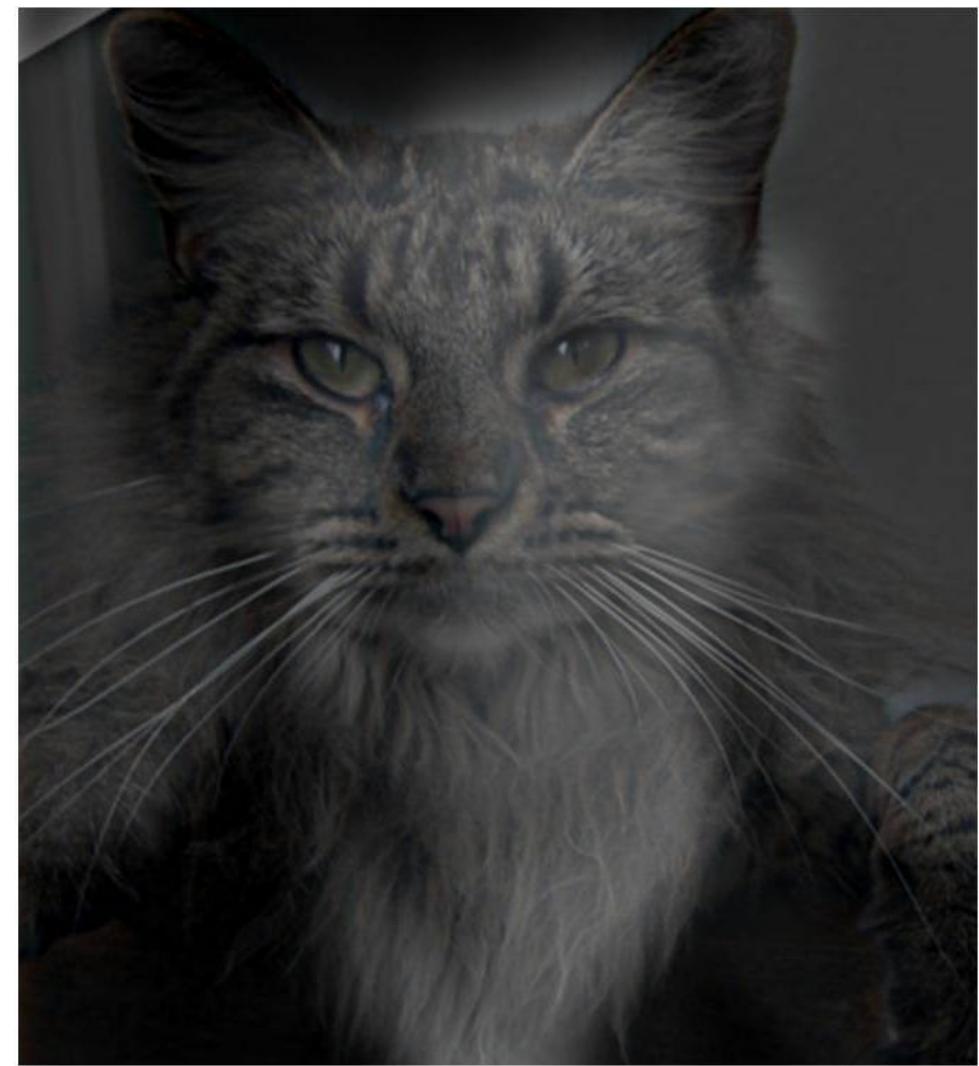
Gaussian



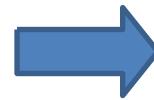
Box filter



# WHY DO WE GET DIFFERENT, DISTANCE-DEPENDENT INTERPRETATIONS OF HYBRID IMAGES?



# WHY DOES A LOWER RESOLUTION IMAGE STILL MAKE SENSE TO US? WHAT DO WE LOSE?



# TODAY'S LECTURE

- Fourier transform and frequency domain
  - Frequency view of filtering
  - Another look at hybrid images
  - Sampling

# JEAN BAPTISTE JOSEPH FOURIER (1768-1830 )

## Idea (1807):

*Any univariate function can be rewritten as a weighted sum of sines and cosines of different frequencies.*

Don't believe it?  
Neither did Lagrange,  
Laplace, Poisson, etc.  
Not translated into English  
until 1878!

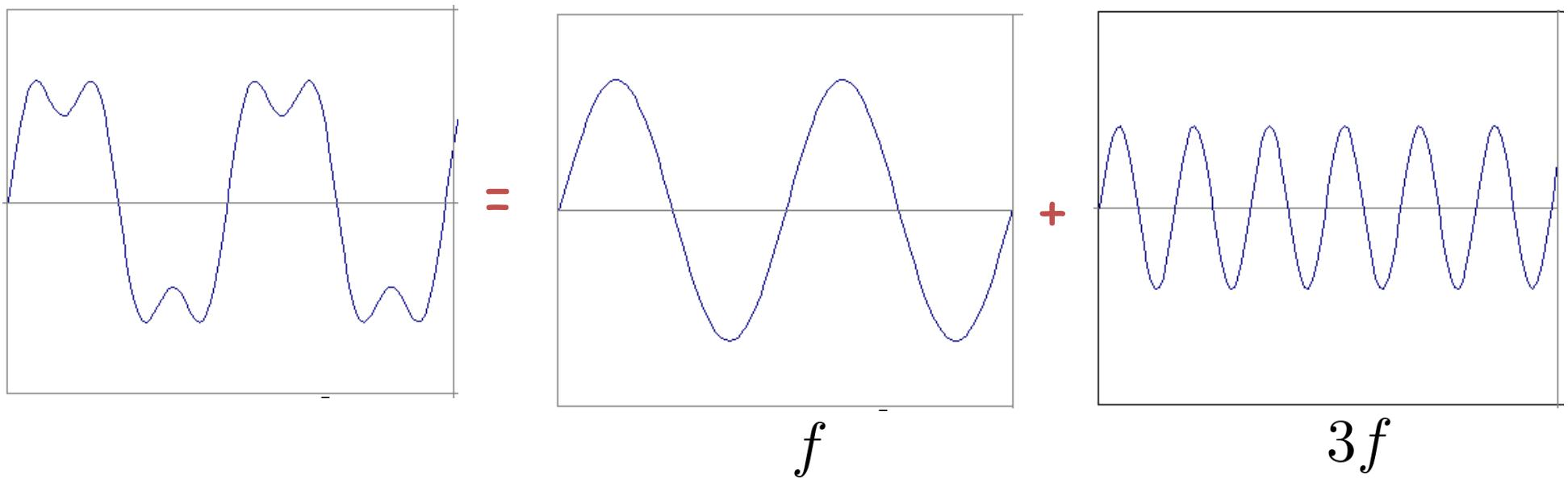
But it's (mostly) true!  
called Fourier Series but  
there are some subtle  
restrictions.



# Fourier analysis 1D

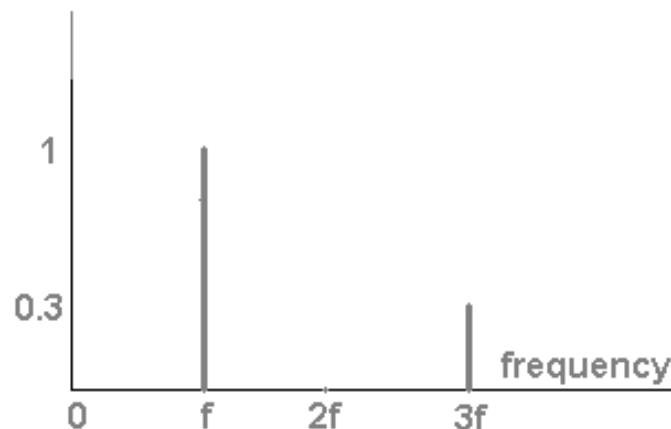
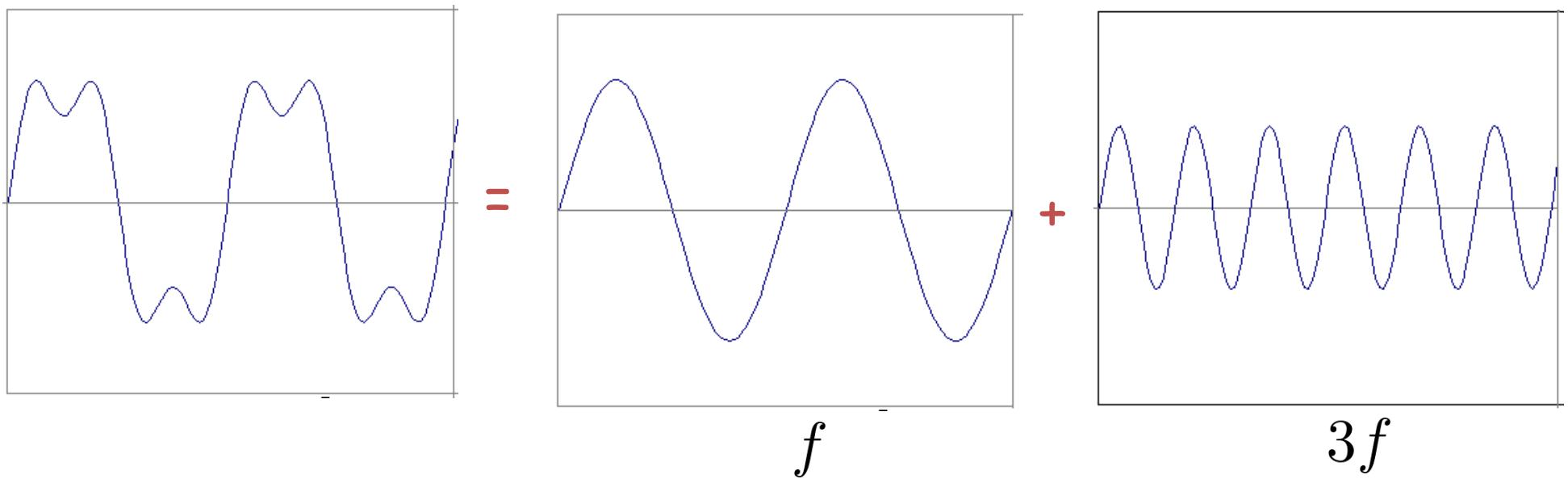
# EXAMPLE

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

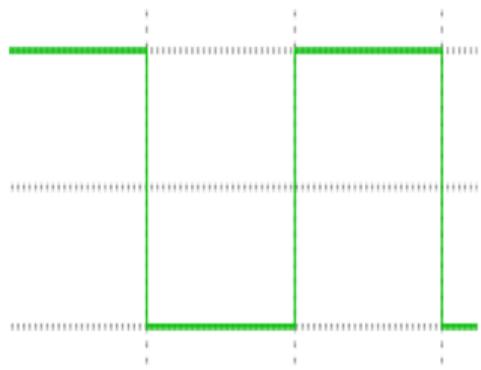


# EXAMPLE

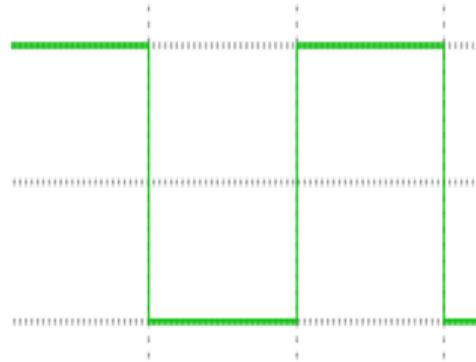
- example :  $g(t) = \sin(2\pi f t) + (1/3)\sin(2\pi(3f) t)$



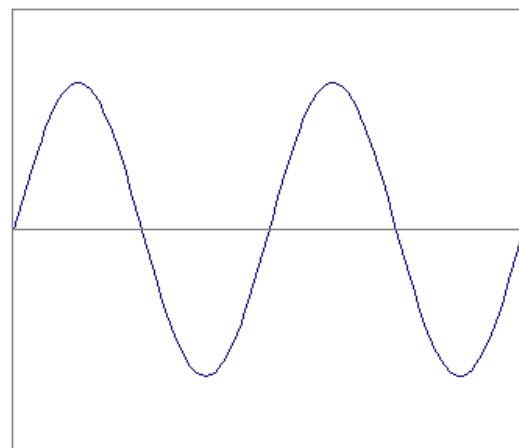
# EXAMPLE



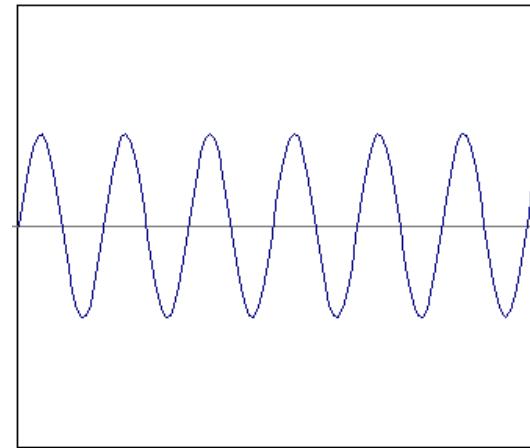
# EXAMPLE



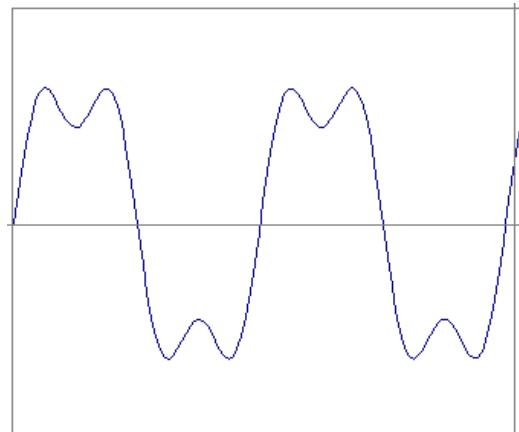
=



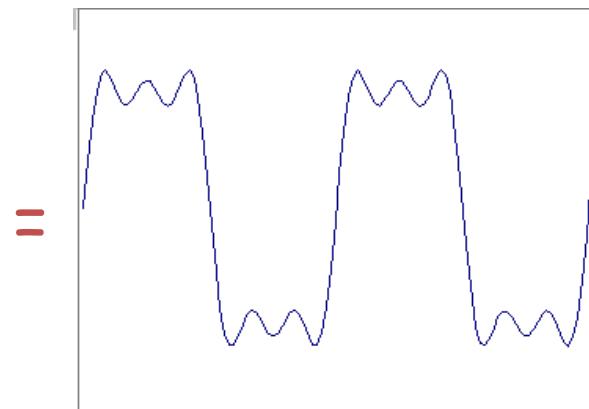
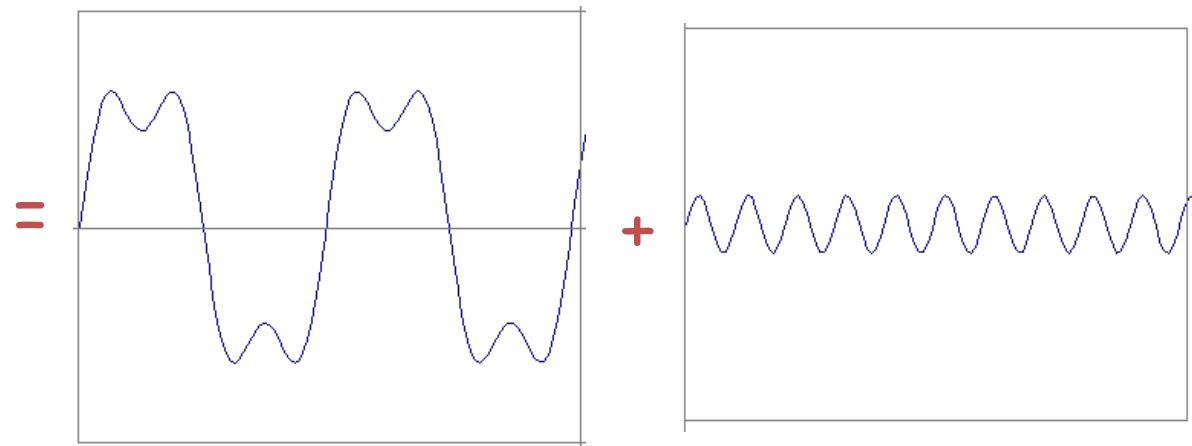
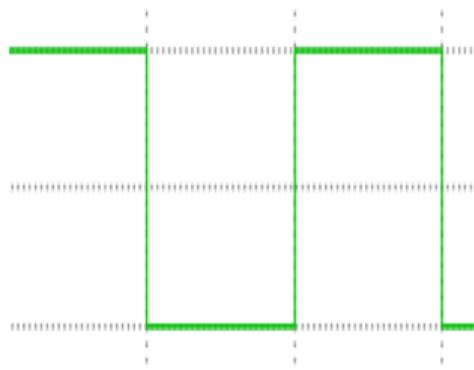
+



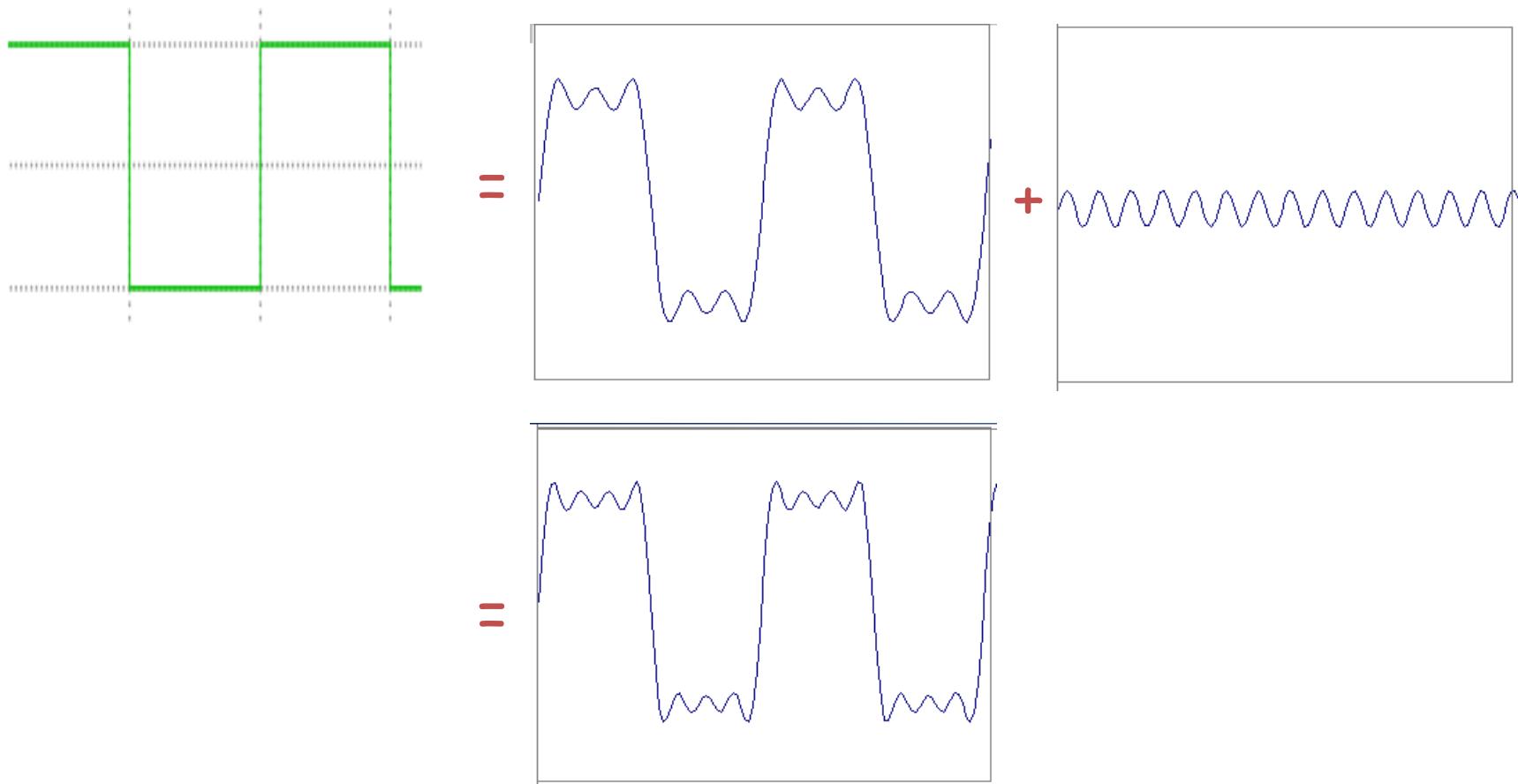
=



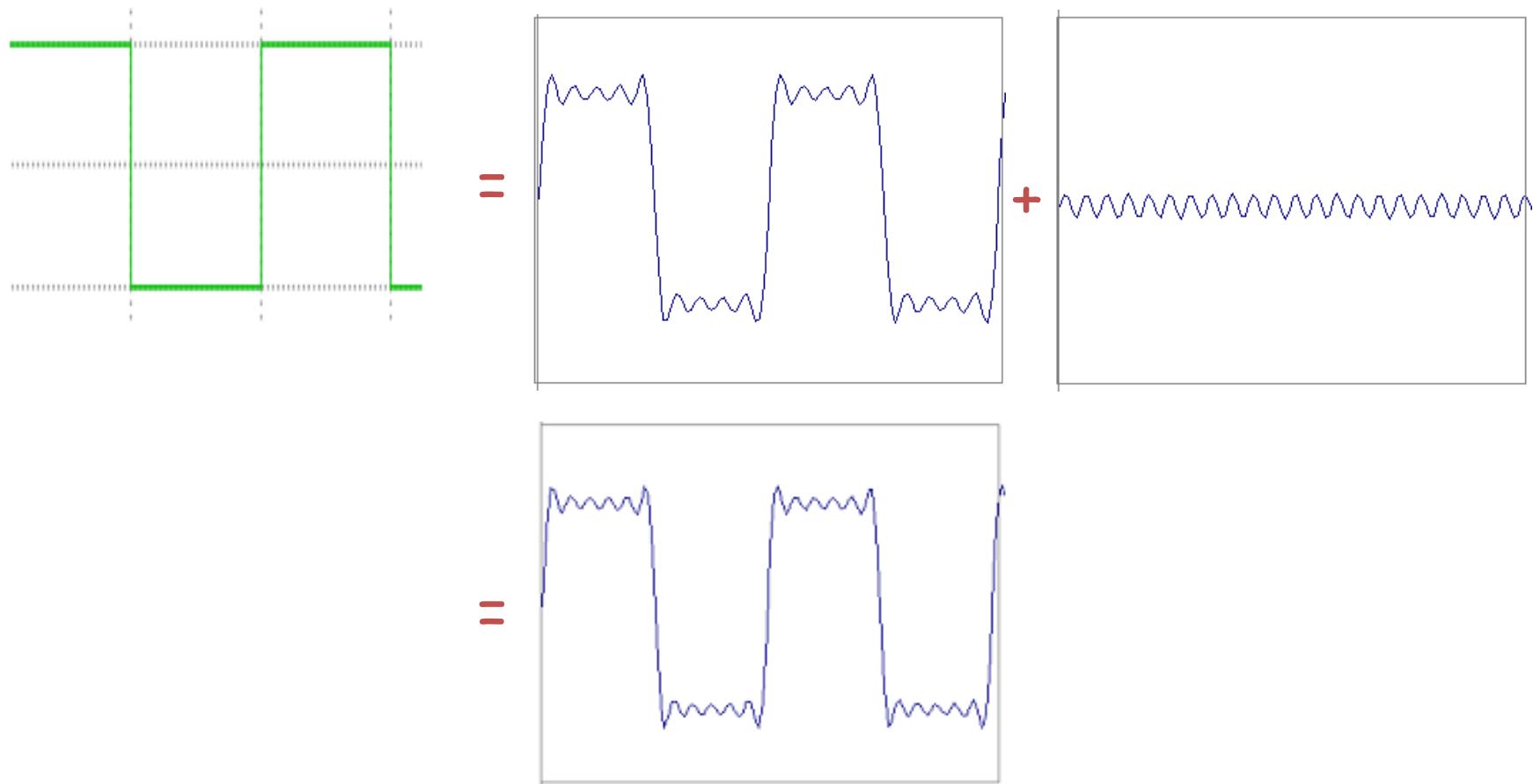
# EXAMPLE



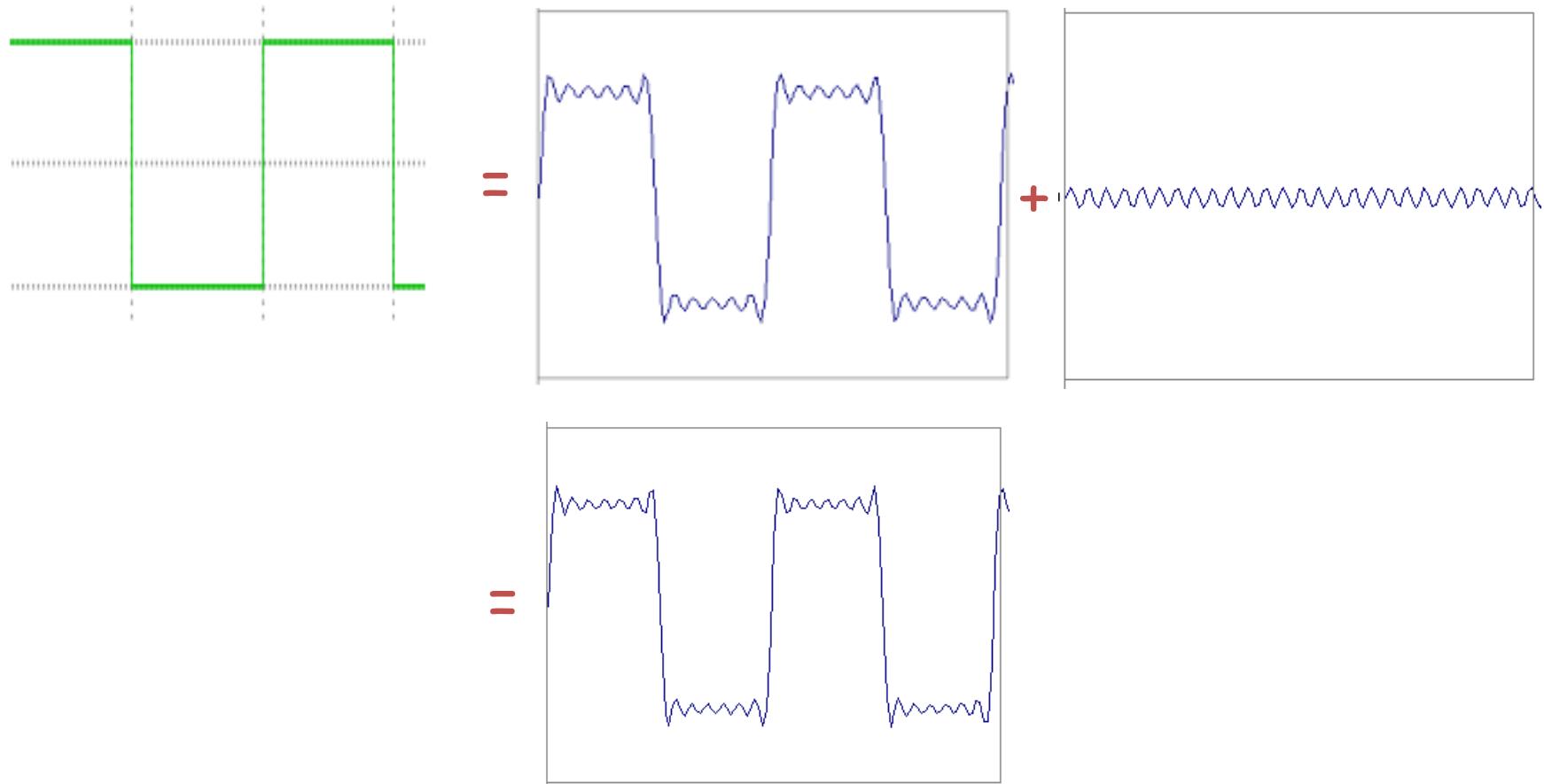
# EXAMPLE



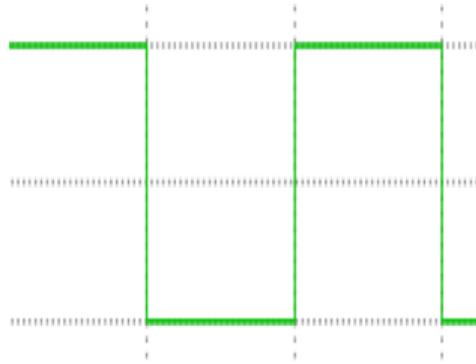
# EXAMPLE



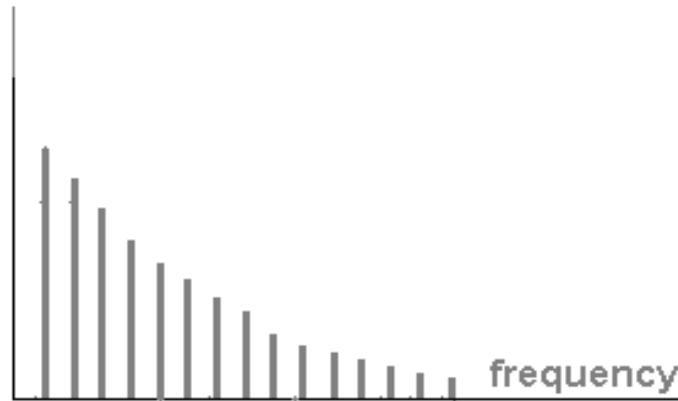
# EXAMPLE



# EXAMPLE



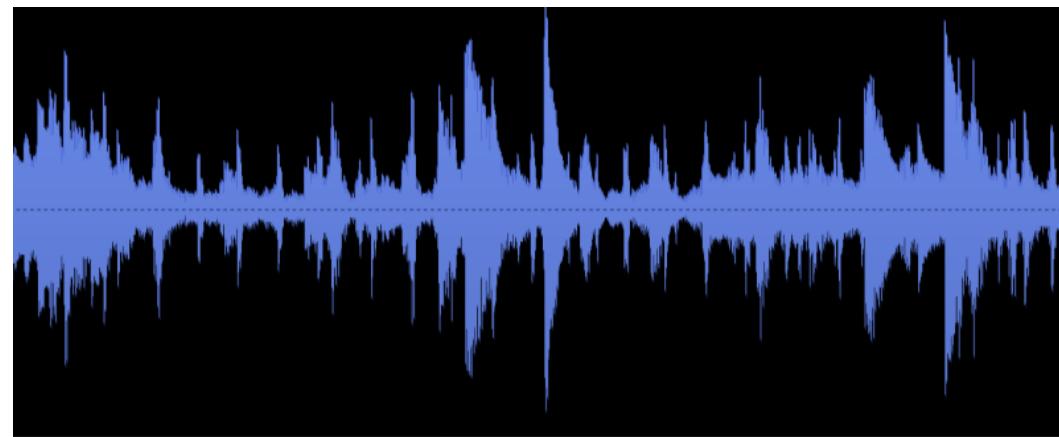
$$= A \sum_{i=1}^{\infty} \frac{1}{k} \sin(2\pi k t)$$



# EXAMPLE

- We think of music in terms of frequencies at different magnitudes

Music  
signal



Frequency  
spectrum



# THE FOURIER SERIES

Fourier series:

**Any even function** can be decomposed as follows

$$C(x) = a_0 + a_1 \cos x + a_2 \cos 2x$$

**Any odd function** can be decomposed as follows

$$S(x) = b_1 \sin x + b_2 \sin 2x$$

**Any function** can be decomposed as follows

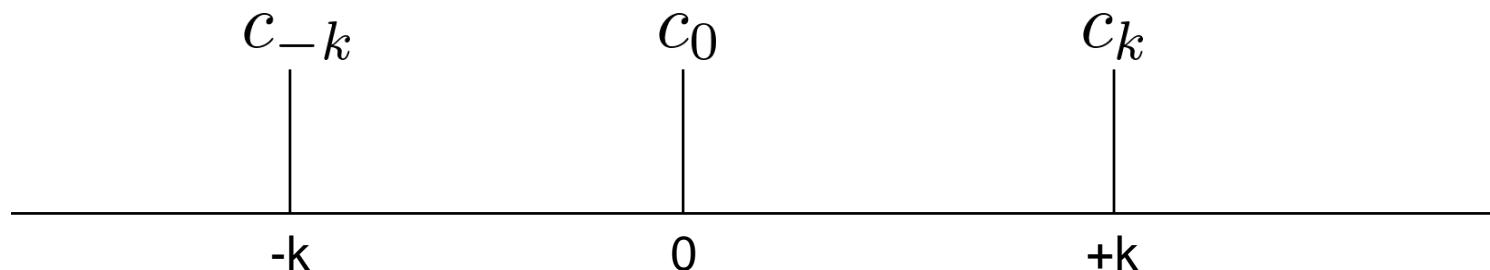
$$F(x) = C(x) + S(x)$$

In compact (cool) notation  $e^{ix} = \cos x + i \sin x$

$$F(x) = \sum_{k=-\infty}^{\infty} c_k e^{ikx}$$

# THE FOURIER TRANSFORM

Take the coefficients and put them in one axis. We obtain the **“frequency spectrum”**.



# THE DISCRETE FOURIER TRANSFORM

A detail: up to this moment we assume functions periodic with period  $2\pi$

The discretization of the former periodic signal in  $N$  samples take values at locations

$$F(n) = f(x)|_{x=\frac{2\pi}{N}n} \quad n = 0, \dots, N-1$$

# THE DISCRETE FOURIER TRANSFORM

A detail: up to this moment we assume functions periodic with period  $2\pi$

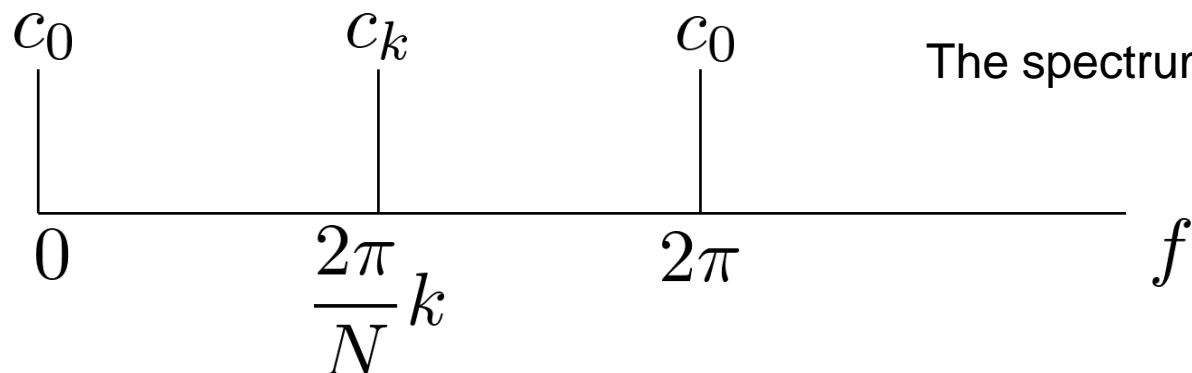
The discretization of the former periodic signal in  $N$  samples take values at locations

$$F(n) = f(x)|_{x=\frac{2\pi}{N}n} \quad n = 0, \dots, N-1$$

Replacing in the continuous case

$$F(x) = \sum_{k=-\infty}^{\infty} c_k e^{ikx} \longrightarrow F(n) = \sum_{k=0}^{N-1} c_k e^{ik\frac{2\pi}{N}n}$$

and the spectrum is defined as



The spectrum is periodic with period  $2\pi$

# A BRIEF NOTE ABOUT THE COEFFICIENTS

$$c_k = \sum_{n=0}^{N-1} f\left(\frac{2\pi}{N}n\right) e^{-ik\frac{2\pi}{N}n}$$

The coefficients are complex numbers. As such, in the Fourier transform we can refer to the magnitude and phase at each frequency:

- Magnitude encodes how much signal there is at a particular frequency (usual value displayed in the spectrum)
- Phase encodes spatial information (indirectly)

$$\mathcal{F}(f(x - x_0)) = F(x)e^{-i\omega x_0}$$

$$|c_k| = \pm \sqrt{R(\omega_k) + I(\omega_k)} \qquad \phi_k = \tan^{-1} \frac{I(w_k)}{R(w_k)}$$

# A BRIEF NOTE ABOUT THE COEFFICIENTS

They can be computed in  $\mathcal{O}(N \log N)$  !!!  
(with some limitations)

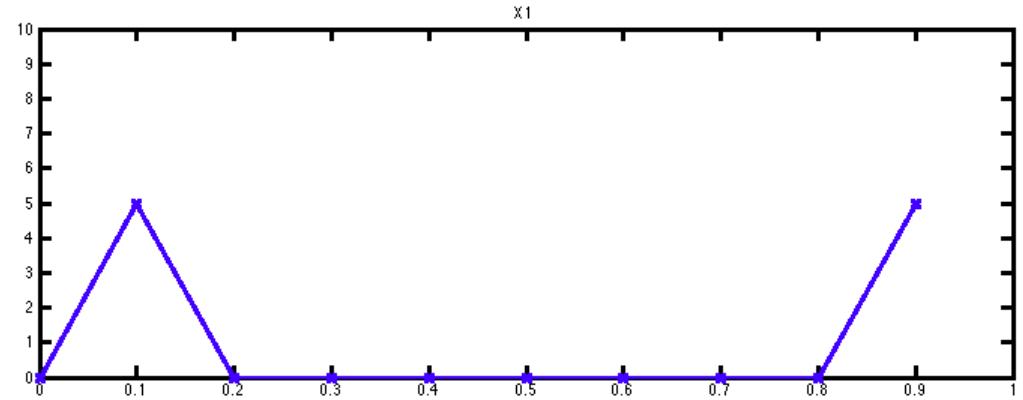
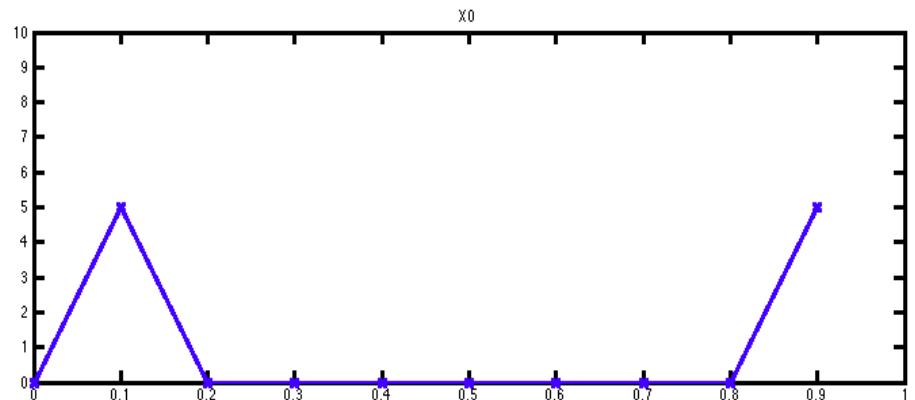
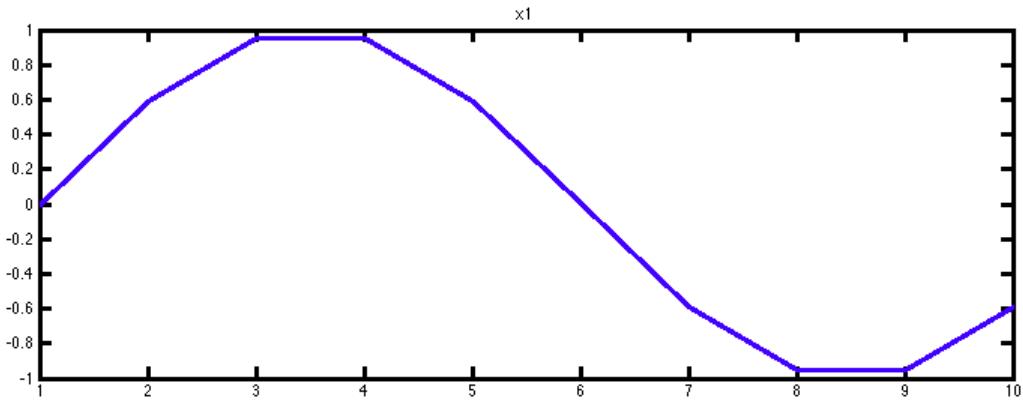
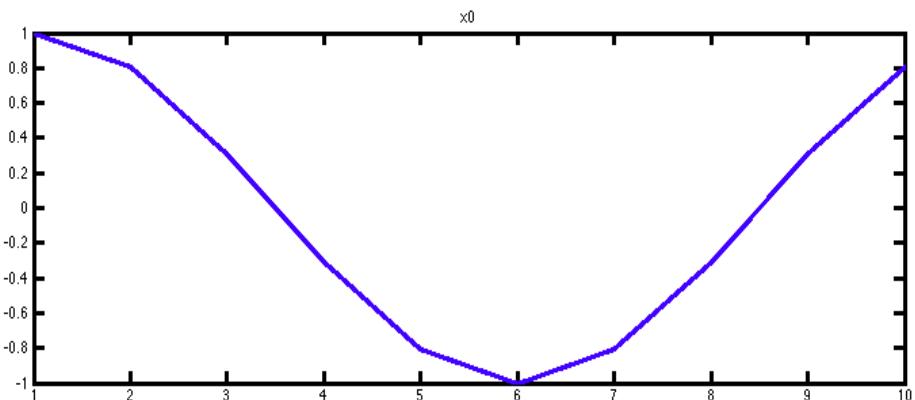
**ENTER THE FAST FOURIER TRANSFORM**

# COFFEE... CODING BREAK 1



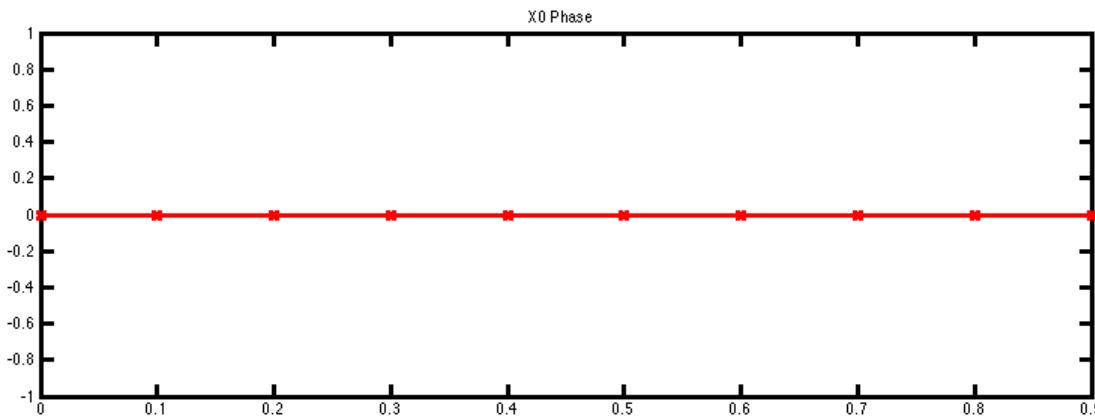
# CODING TIME: sine and cosine transforms magnitude

```
n = [0:9];  
x0 = cos(2*pi*n/10); % 1 period  
N0 = 10;  
X0 = fft(x0,N0);  
plot(abs(X0))
```

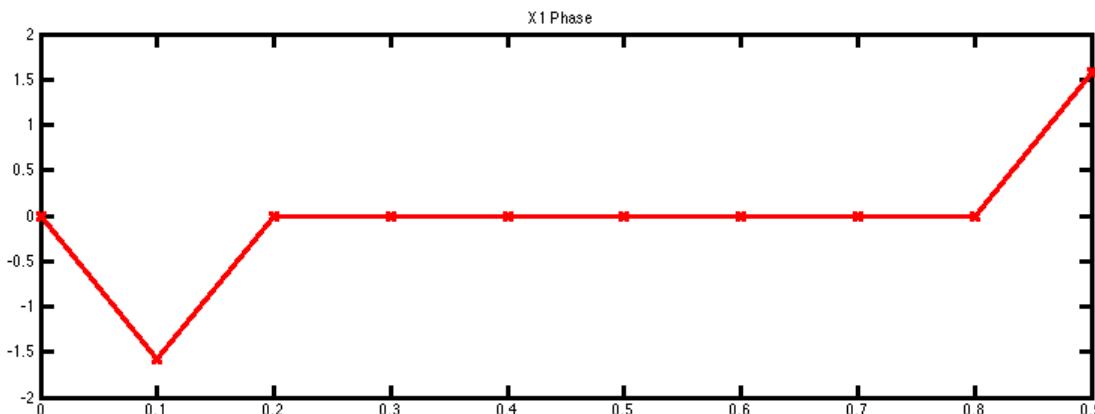


# CODING TIME: sine and cosine transforms phases

```
plot(angle(X0));
```



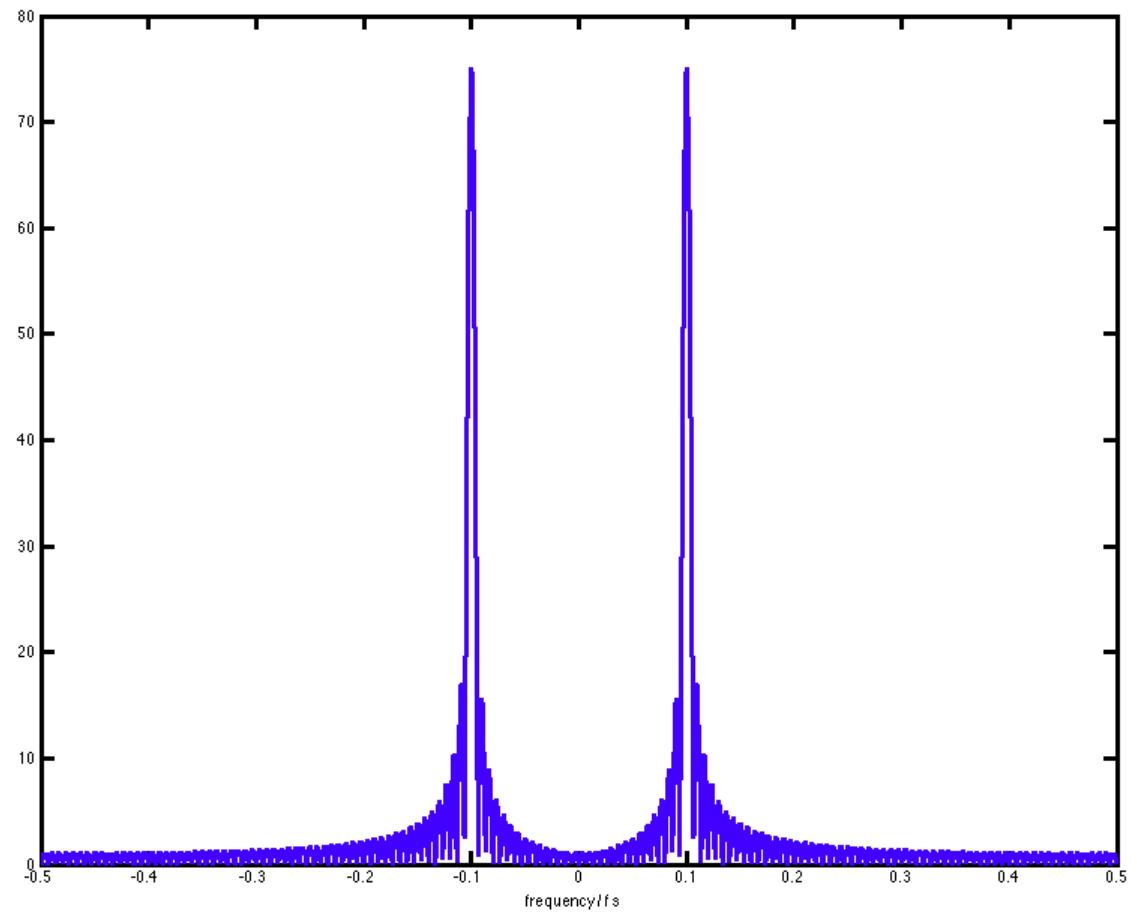
cosine



sine

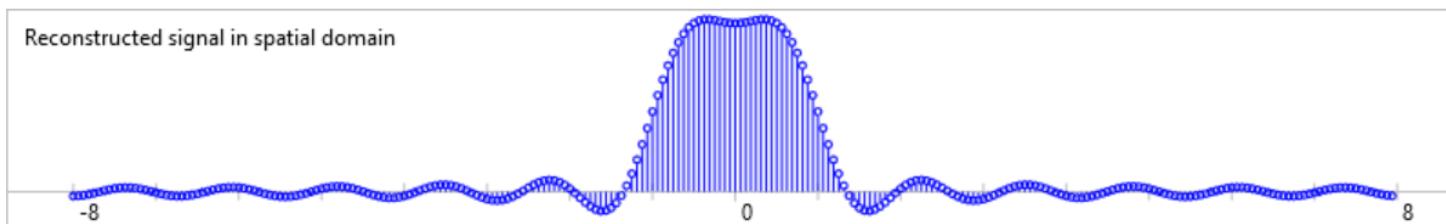
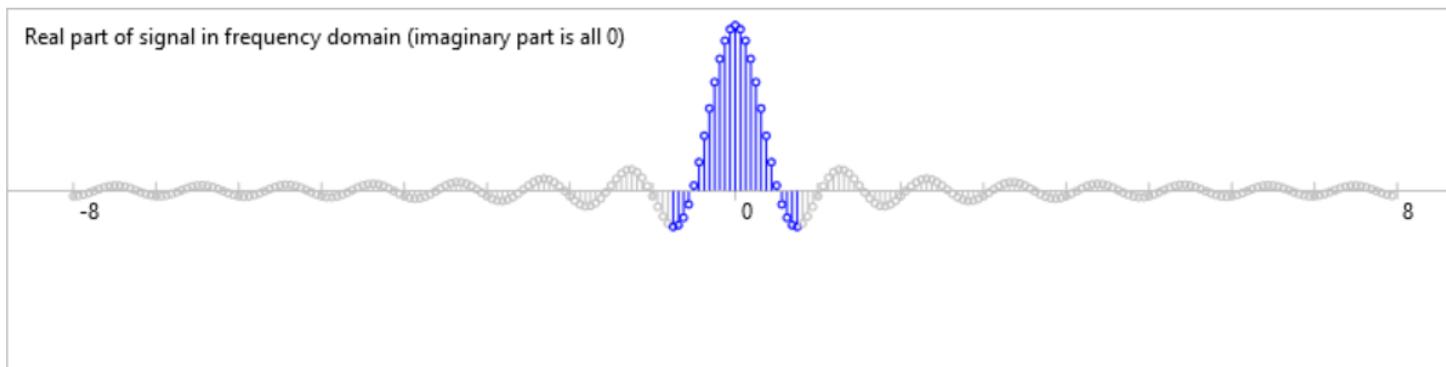
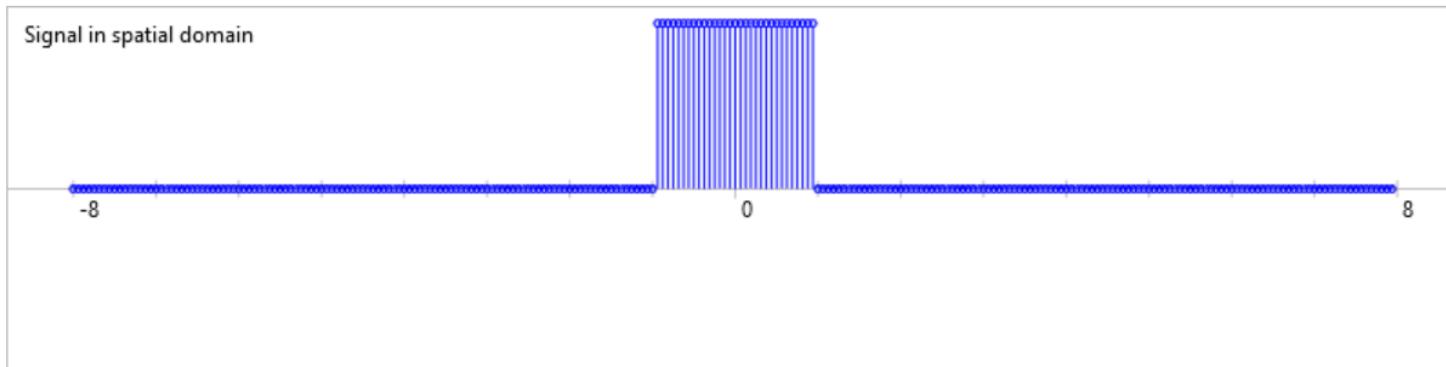
# CODING TIME: symmetry and visualization

```
n = [0:149];
x = cos(2*pi*n/10);
N = 2048;
X = abs(fft(x,N));
X = fftshift(X);
F = [-N/2:N/2-1]/N;
plot(F,X),
```

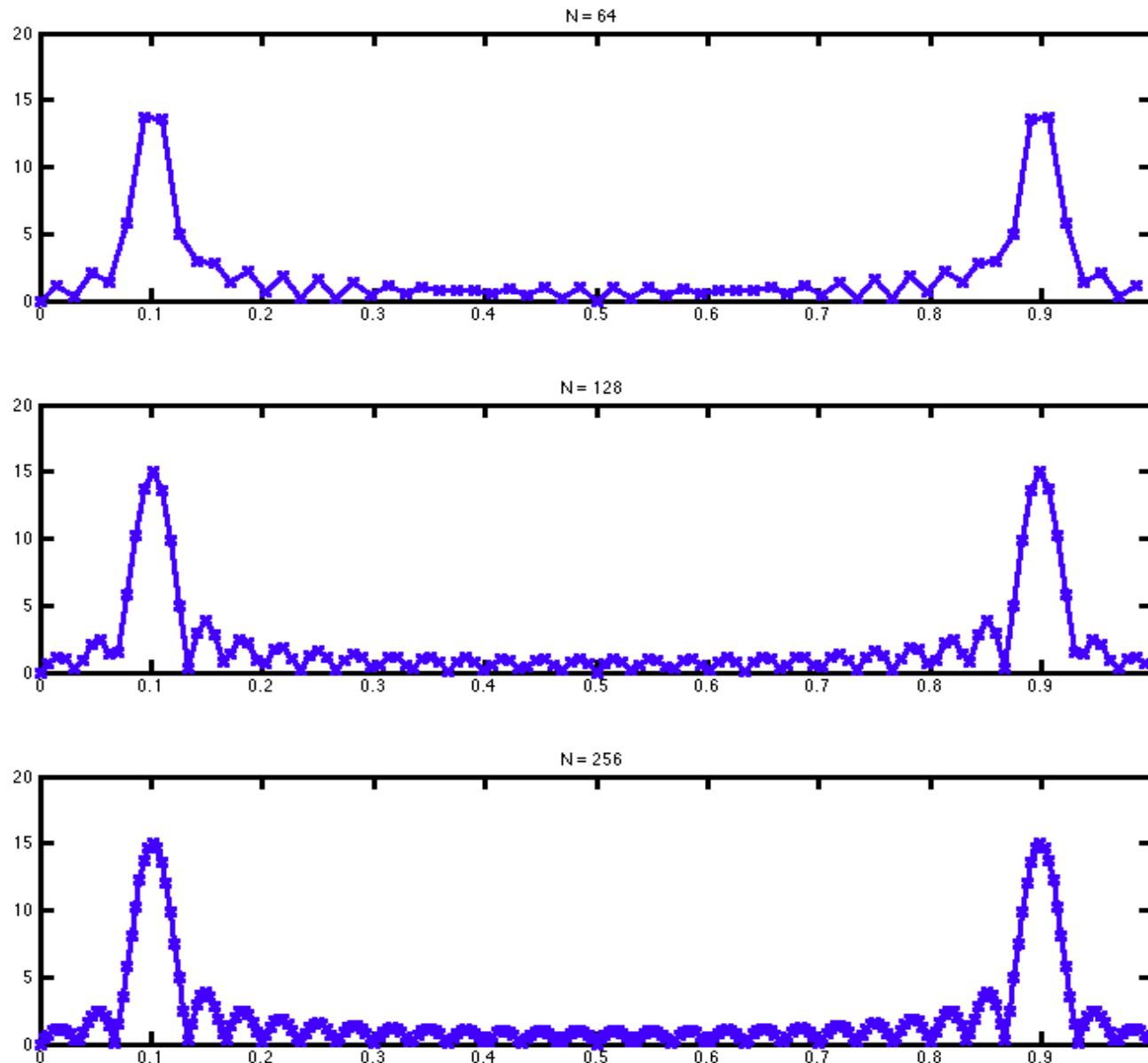


Interesting Live demo:  
<http://madebyevan.com/dft/>

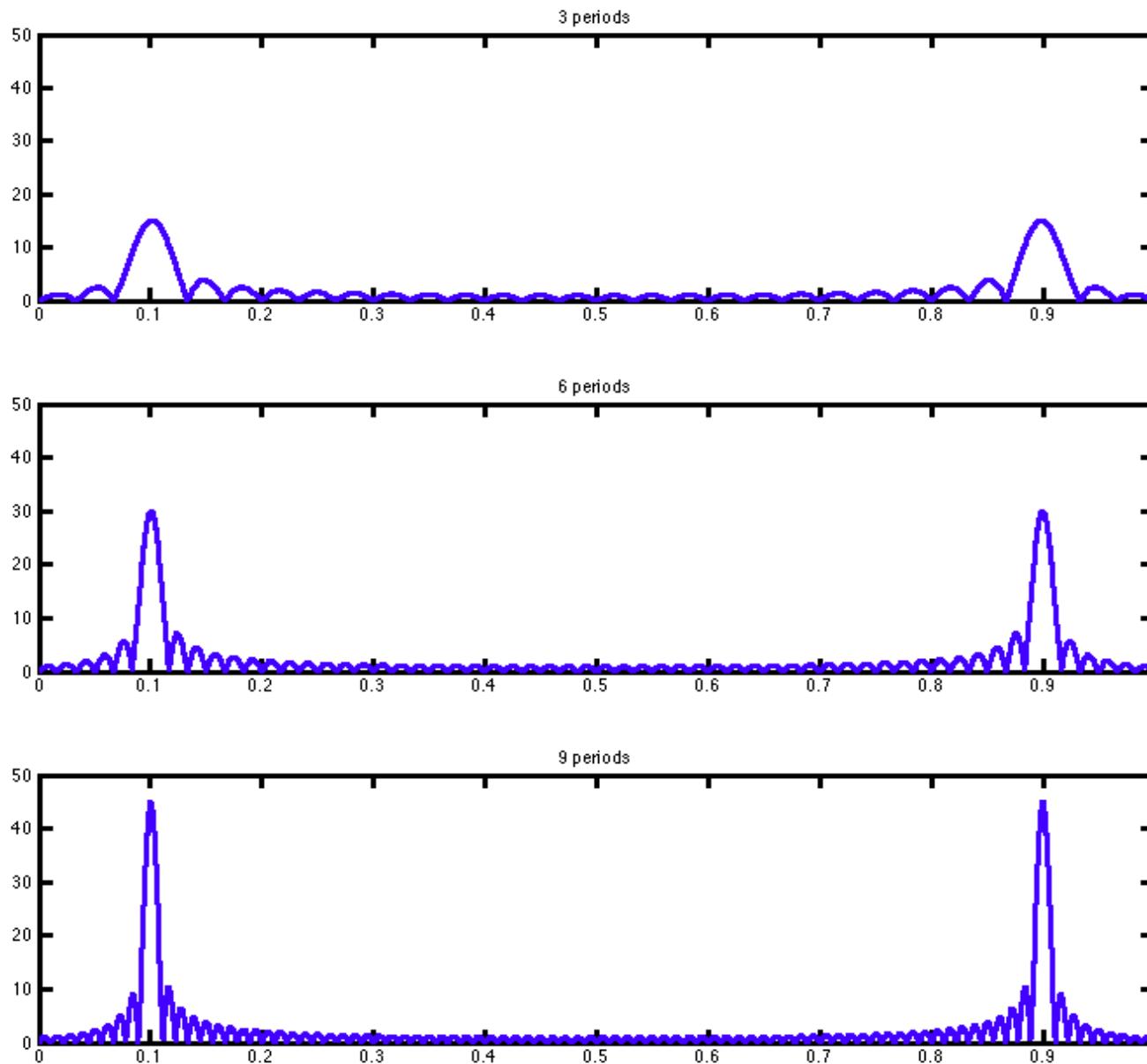
$$f(x) = \boxed{\text{box}(x)}$$



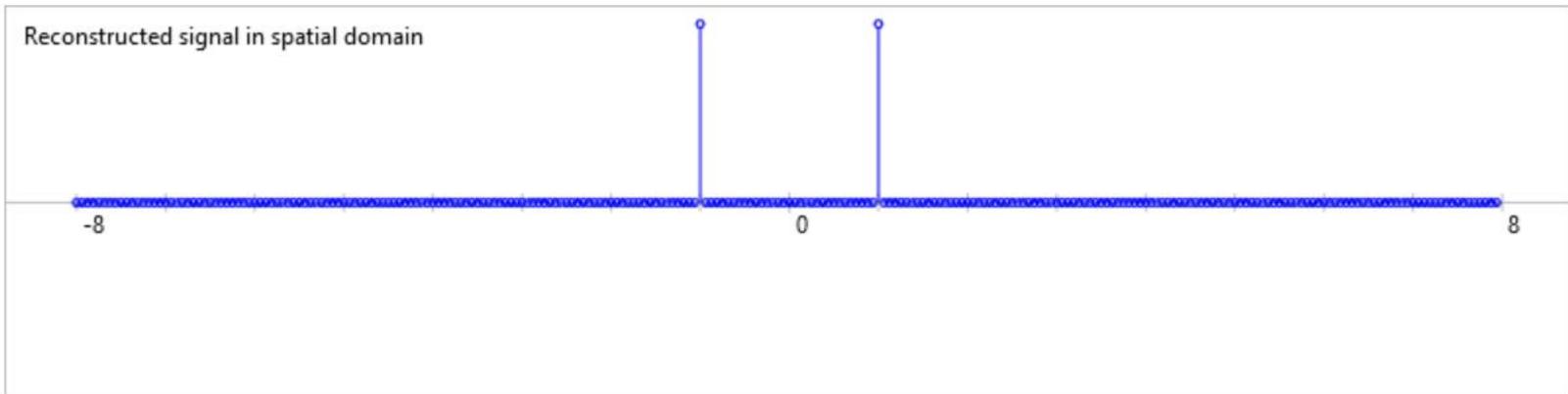
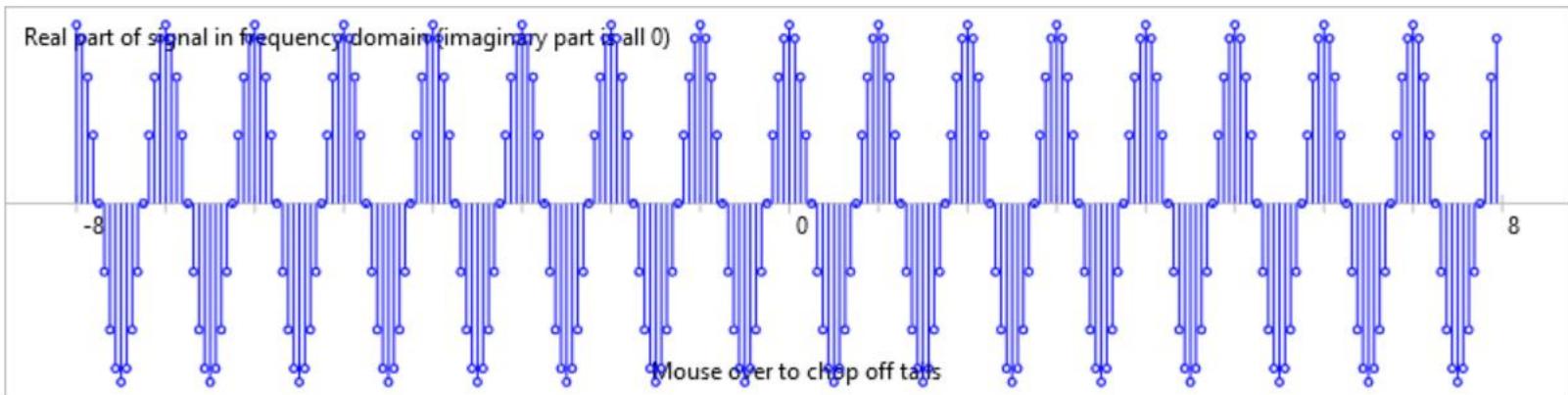
# CODING TIME: size effect



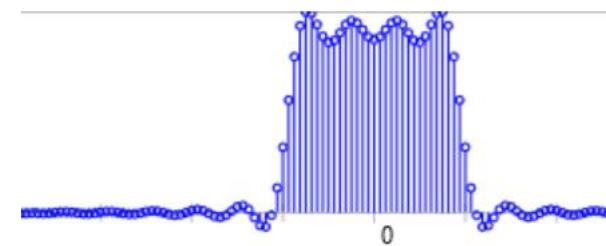
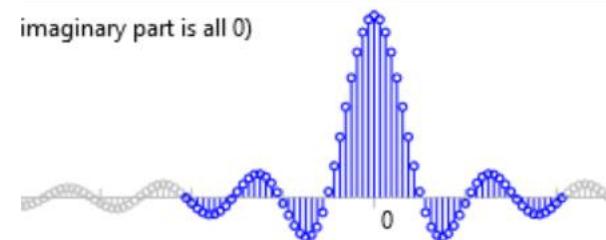
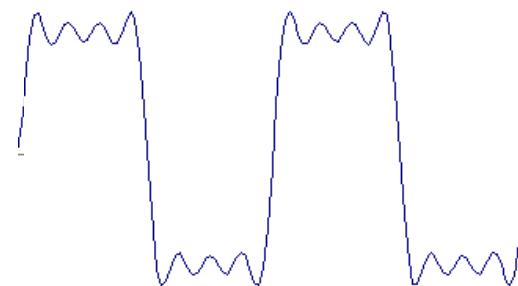
# Signal length effect



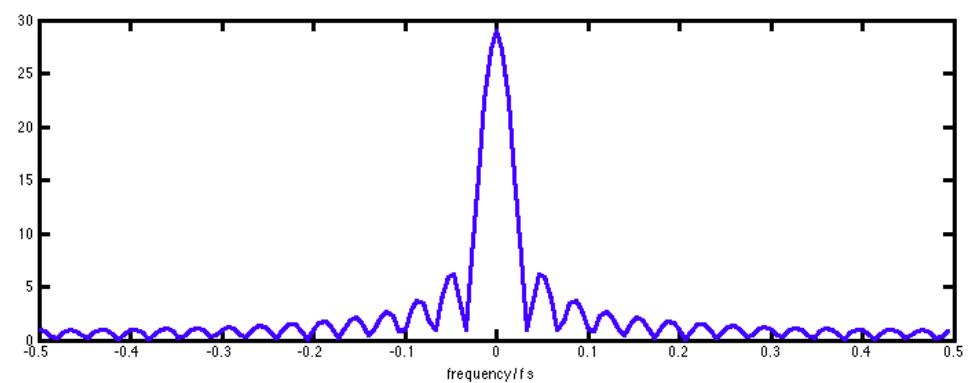
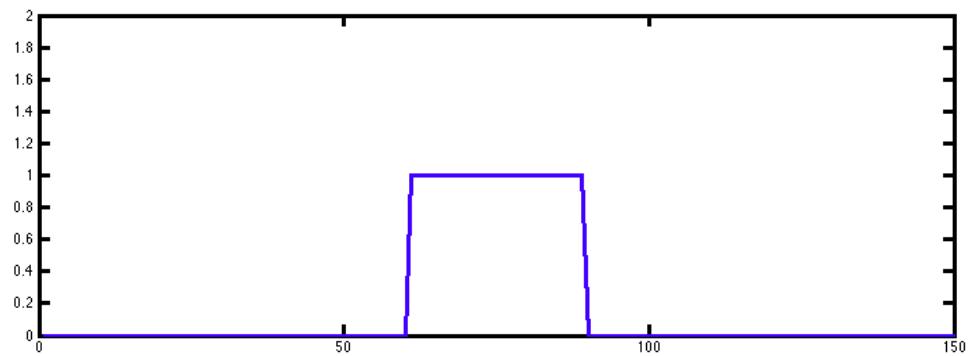
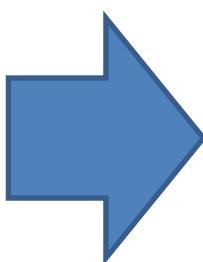
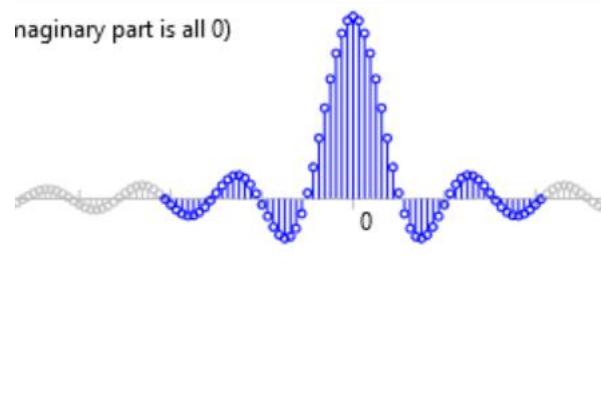
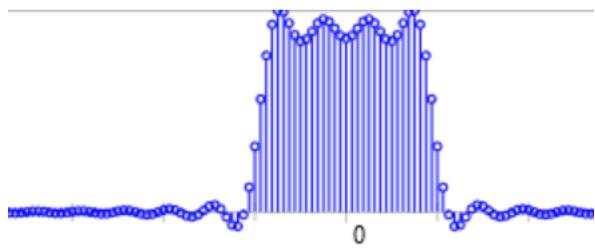
# Signal length effect



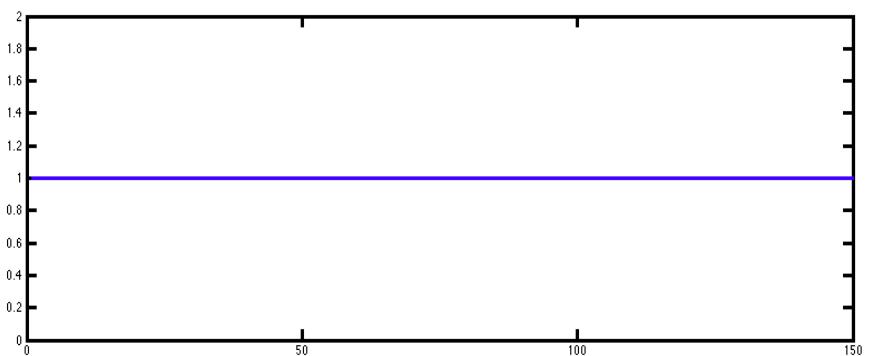
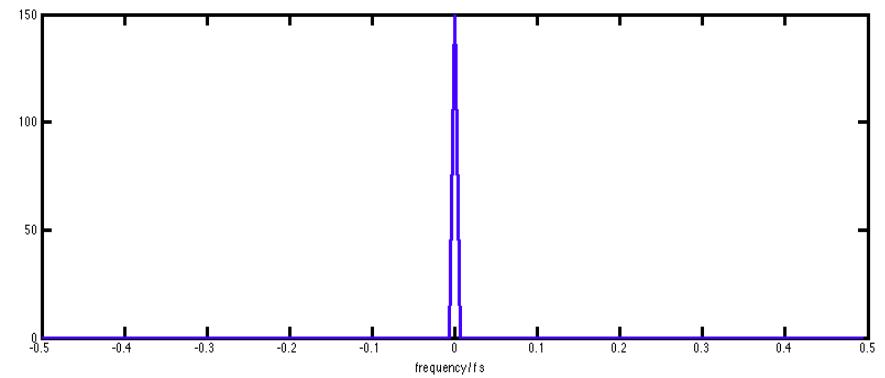
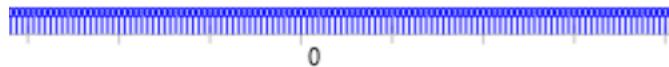
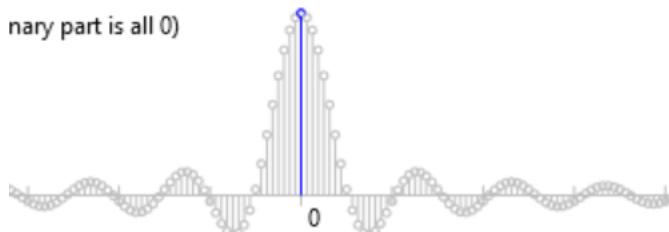
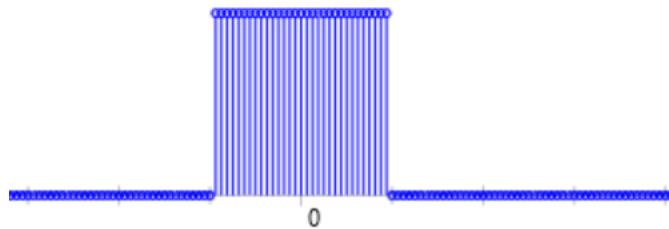
# Approximation of box with sinusoids



# Box transform



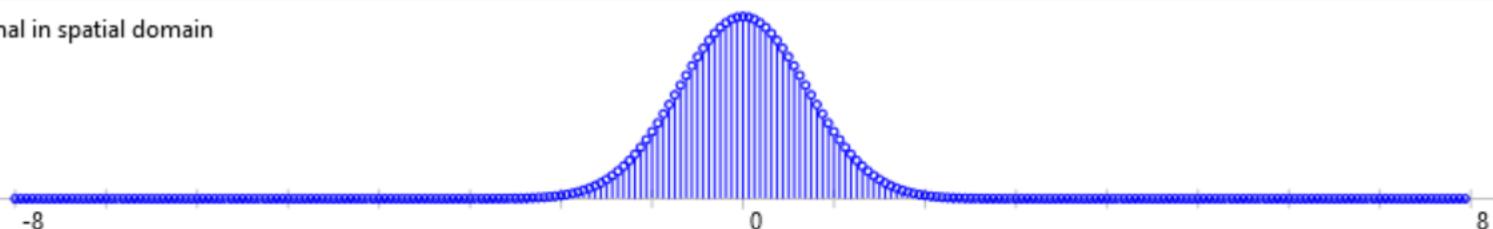
# Transforms of a constant



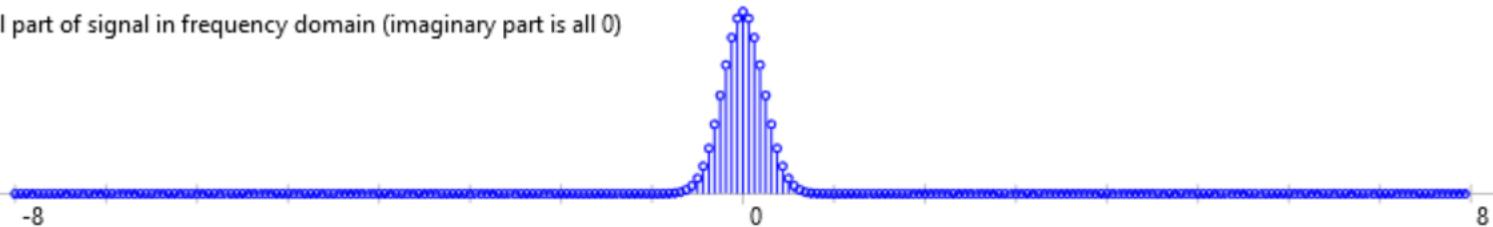
# Transforms of a gaussian

$$f(x) = \text{gaussian}(x)$$

Signal in spatial domain

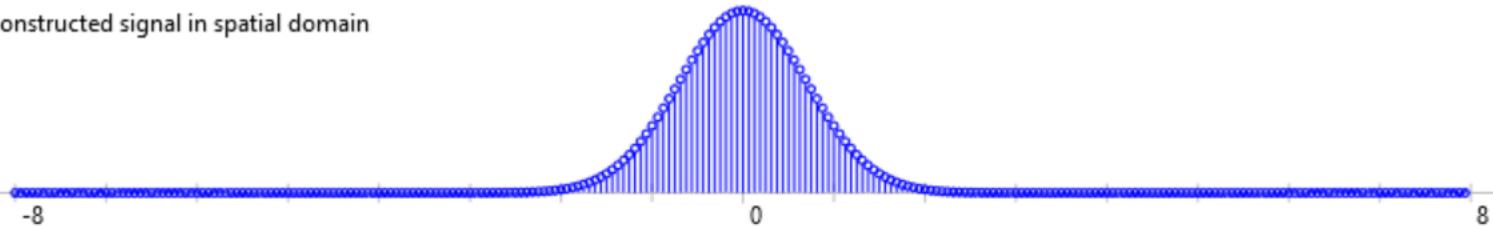


Real part of signal in frequency domain (imaginary part is all 0)



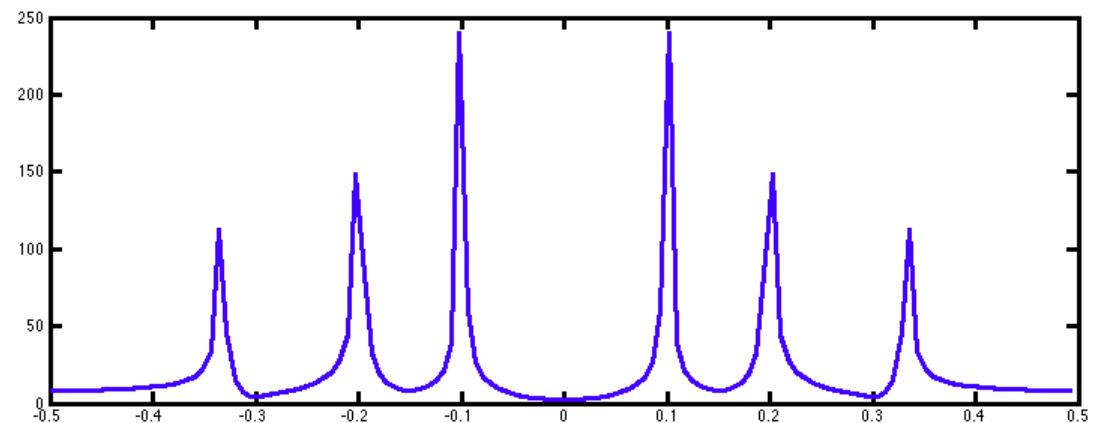
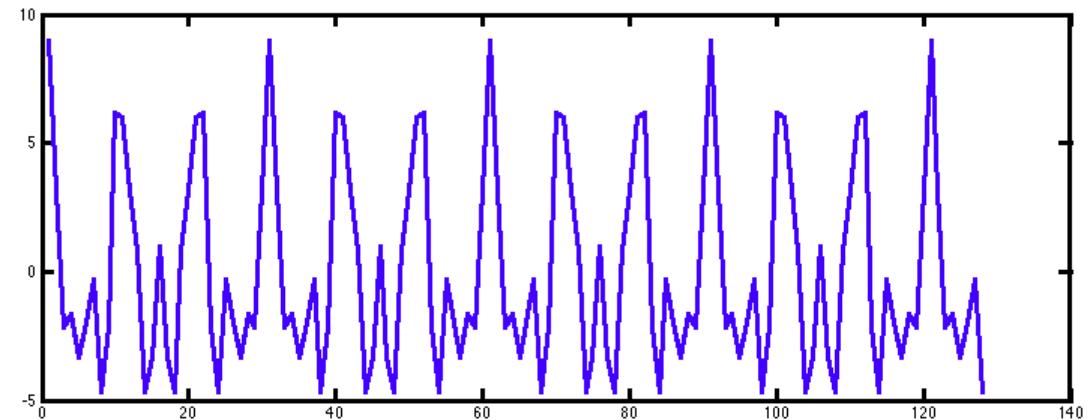
Mouse over to chop off tails

Reconstructed signal in spatial domain

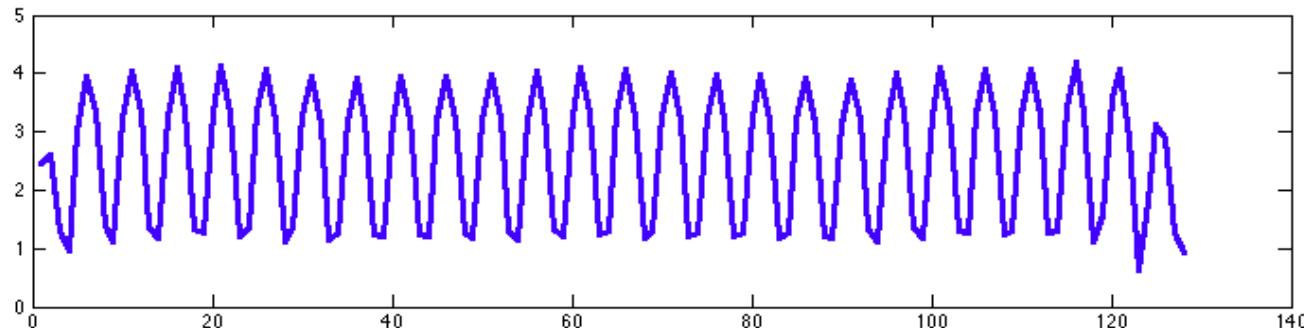
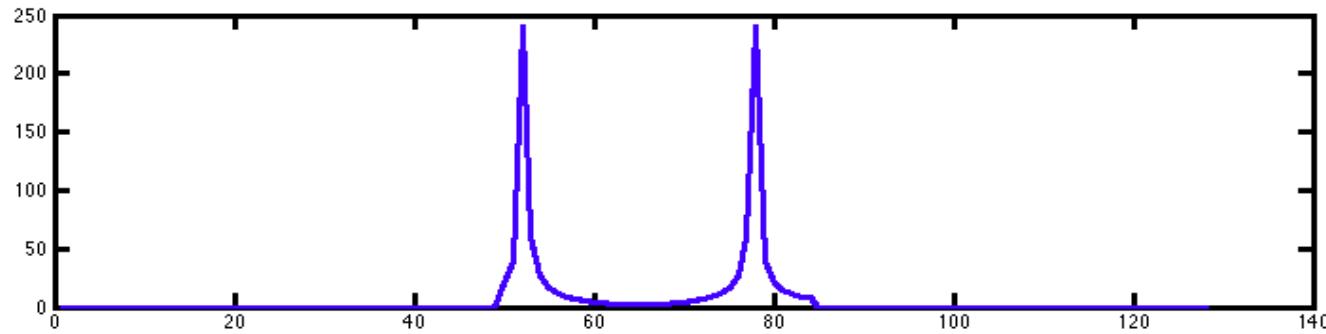
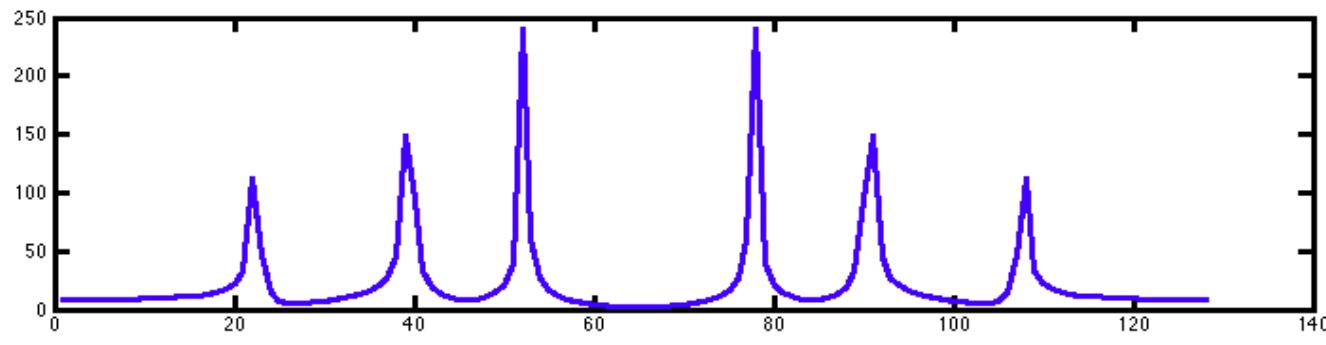


# CODING TIME: composed signal

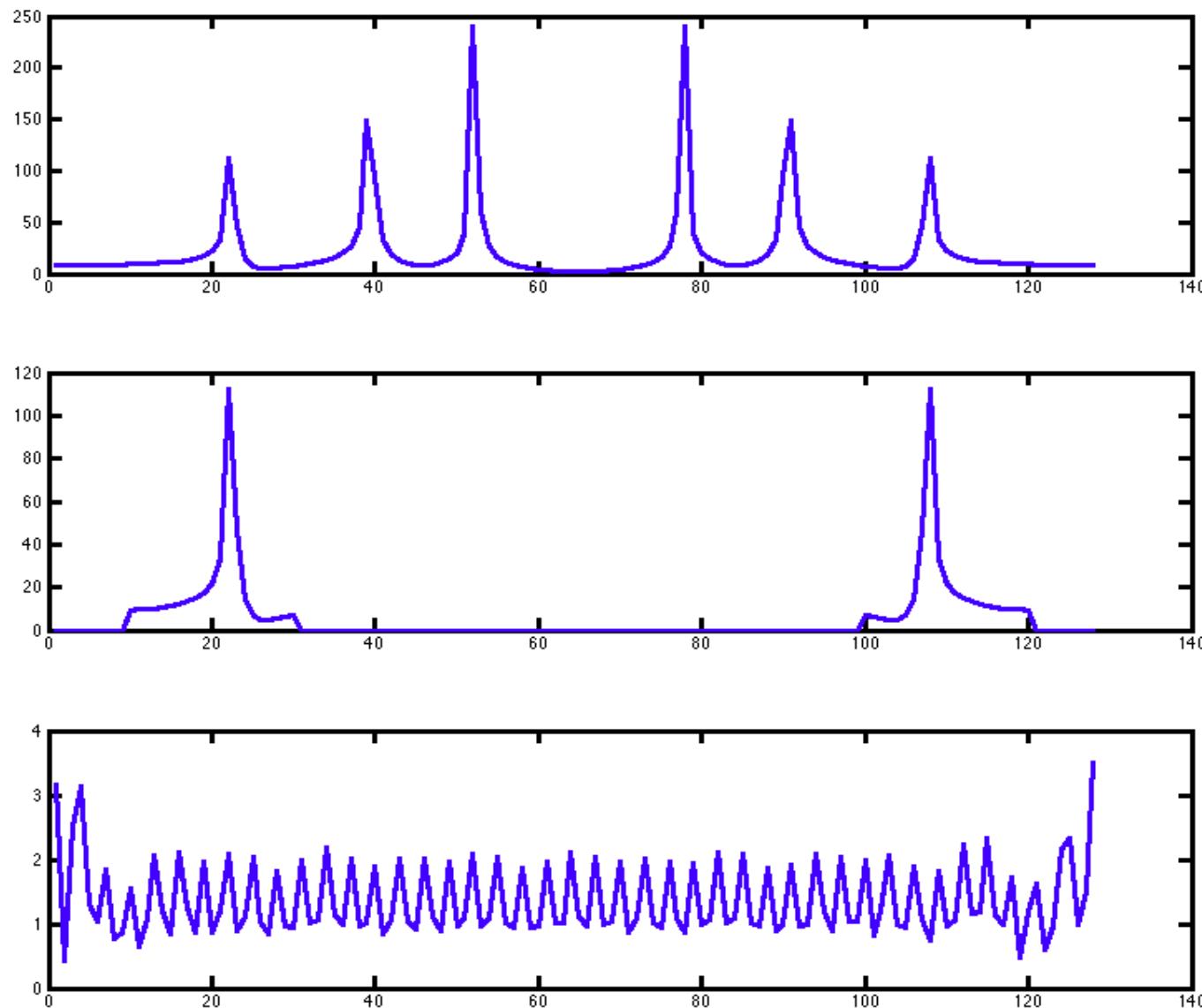
```
n = [0:127];  
x1 = 4*cos(2*pi*n/10); % f=1/10  
x2 = 3*cos(2*pi*n/5); % f=1/5  
x3 = 2*cos(2*pi*n/3); % f=1/3  
x= x1+x2+x3;
```



# CODING TIME: low frequency selection



# CODING TIME: high frequency selection



# APPLICATIONS AND PROPERTIES OF THE FOURIER TRANSFORM

COMMUNICATIONS: Because of this your cellphone, TV, radio, satellite communications, etc work.

$$\mathcal{F}(f(x)e^{i2\pi x w_0}) = F(w - w_0)$$

MATHEMATICAL MODELLING: EDP numerical solution.

$$\mathcal{F}(f'(x)) = i\omega F(\omega)$$

SIGNAL PROCESSING: A fast way of applying a convolution.

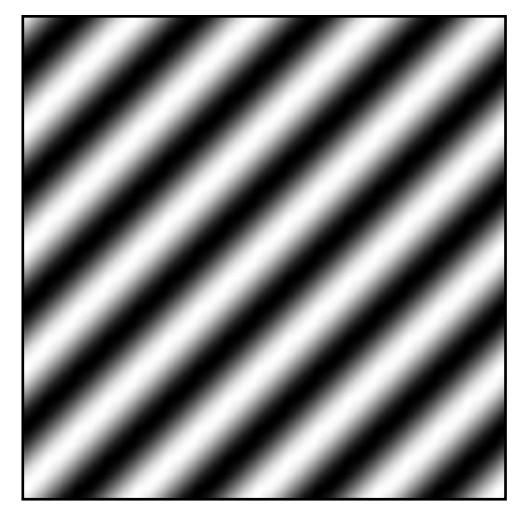
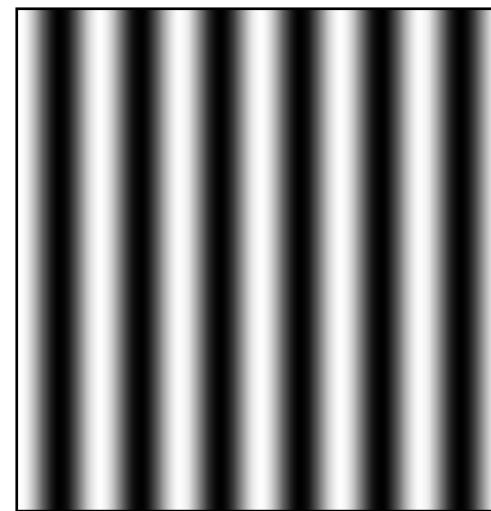
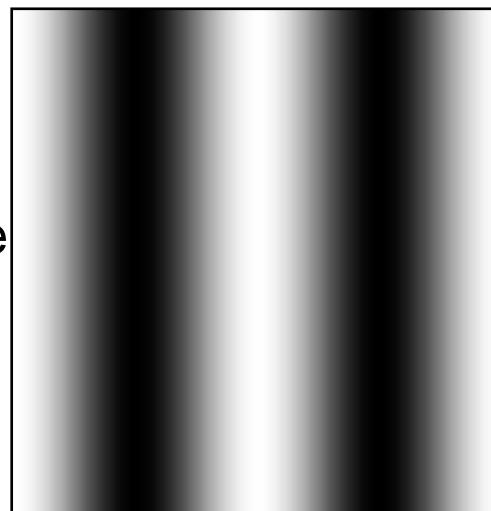
$$\mathcal{F}(f(x) \otimes g(x)) = F(\omega) \cdot G(\omega)$$

SIGNAL ANALYSIS: Interpretation in term of frequencies.

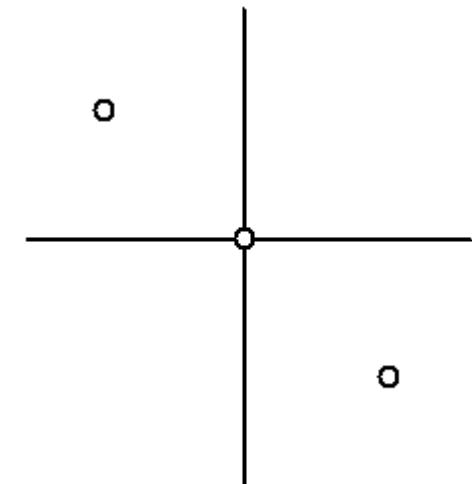
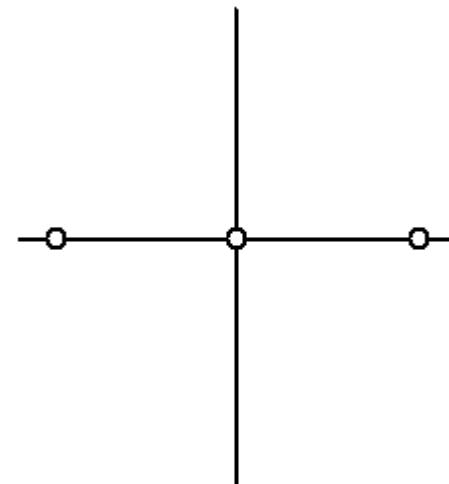
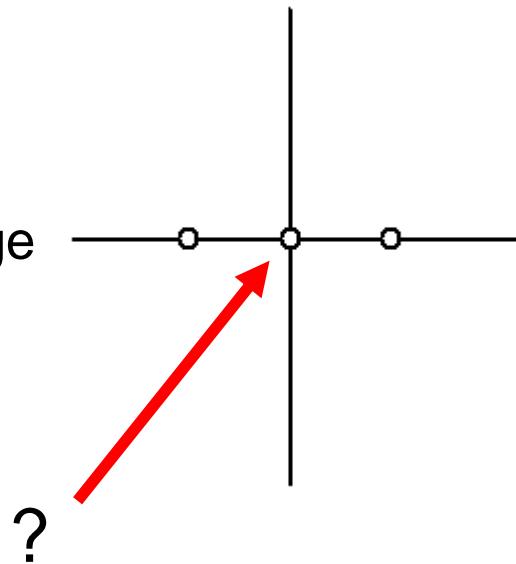
# Fourier analysis 2D

# FOURIER ANALYSIS IN IMAGES

Intensity Image

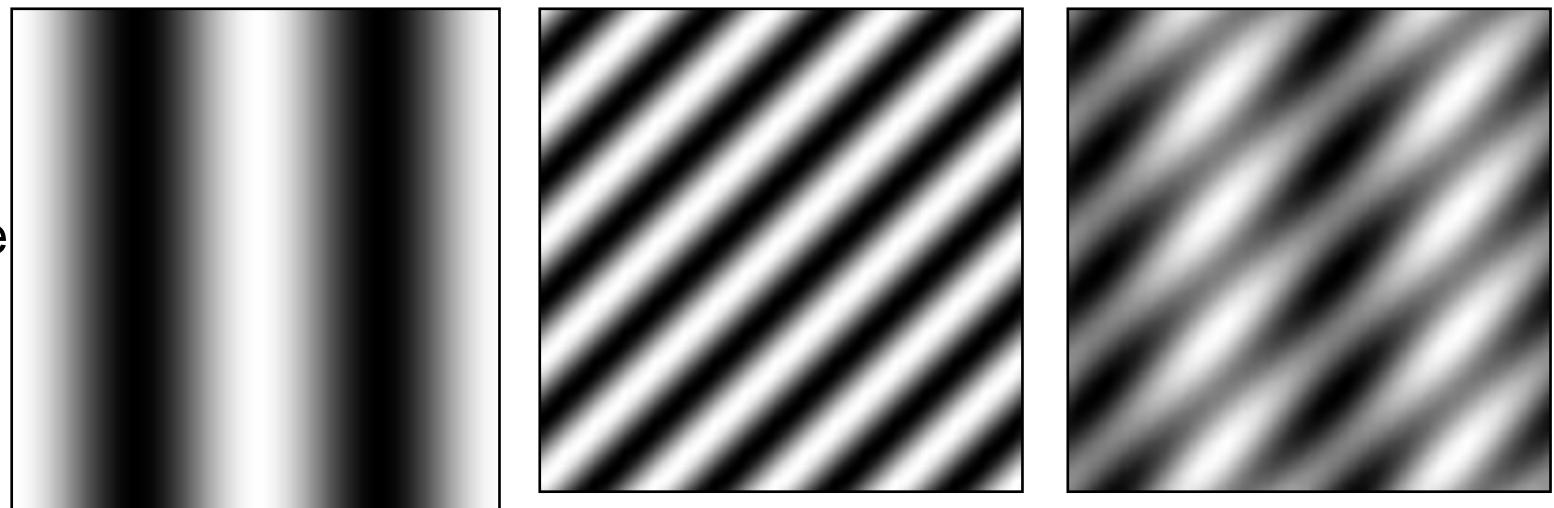


Fourier Image

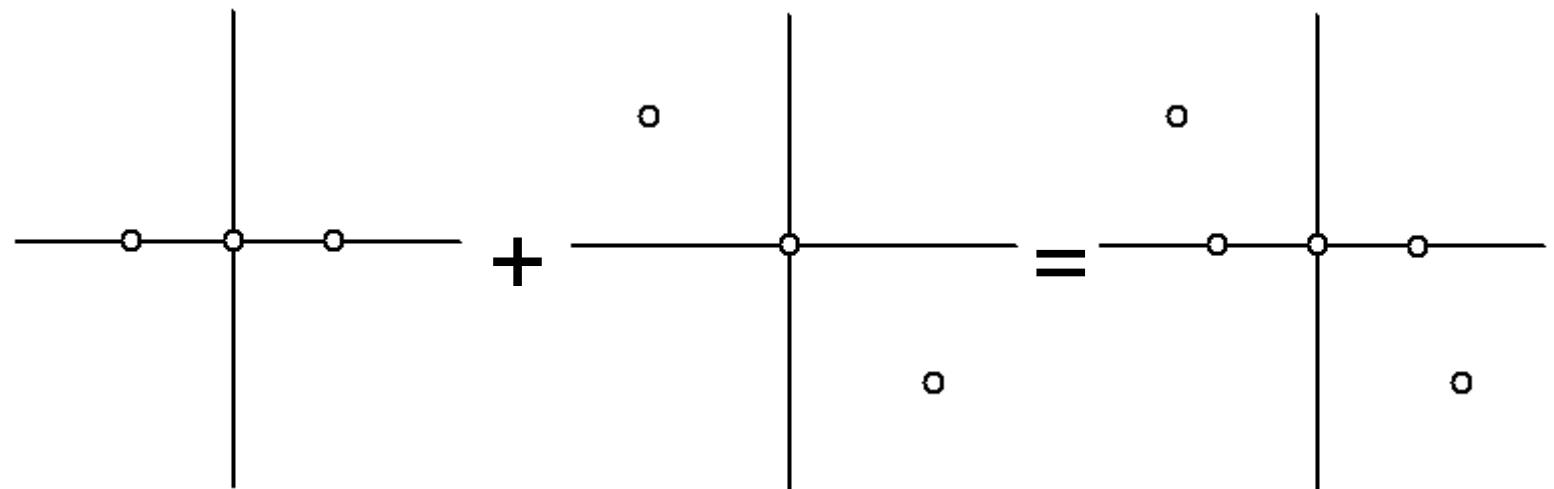


# FOURIER ANALYSIS IN IMAGES

Intensity Image



Fourier Image

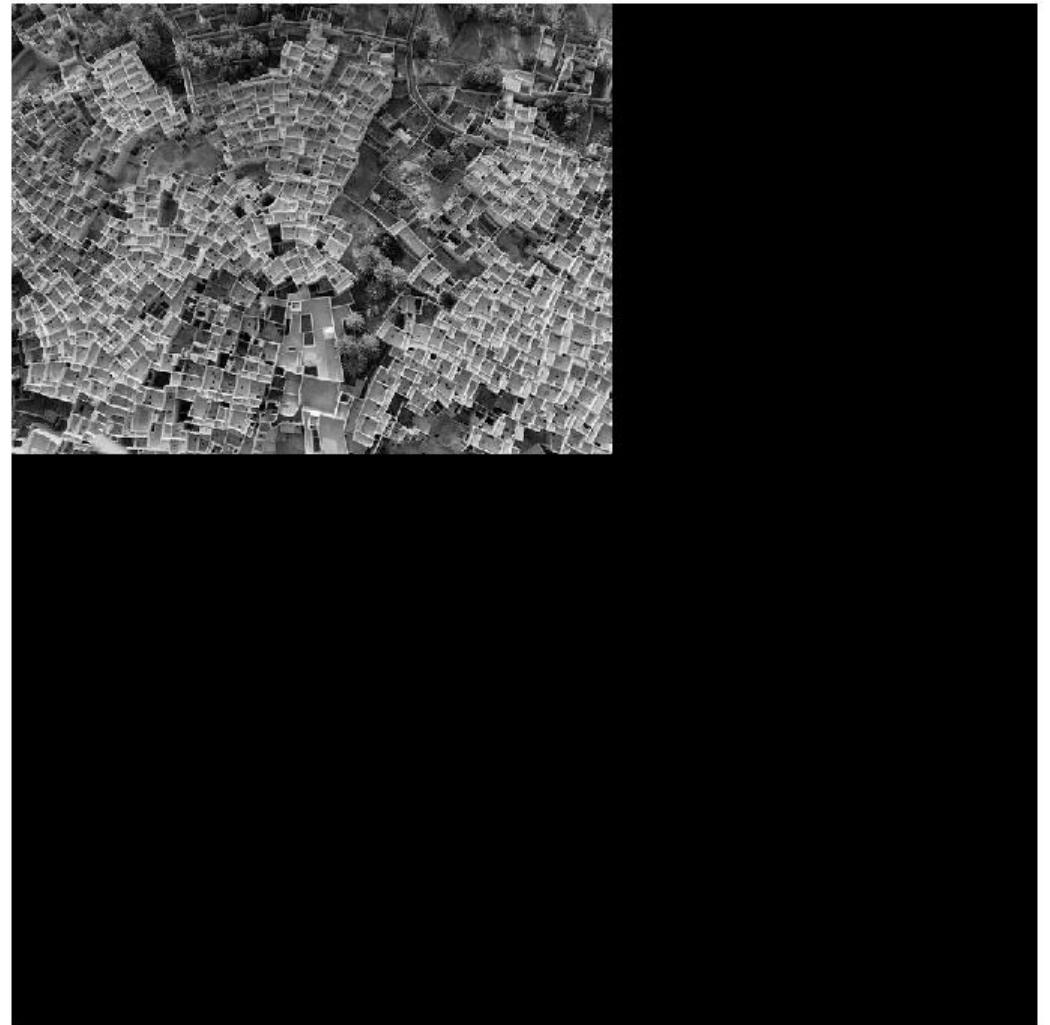


# COFFEE... CODING BREAK 2



# CODING TIME: Fourier transform in 2D

```
fftsize = 1024; % should be order of 2 (for speed) and include padding  
im_fft = fft2(im, fftsize, fftsize);  
disp('Observe the inverse fft to see the padding');  
im_rec = ifft2(im_fft);
```



# CODING TIME: Magnitude and Phase in Fourier

Imagen  
original



Magnitude

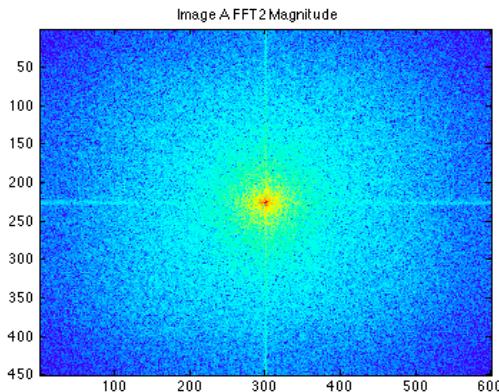
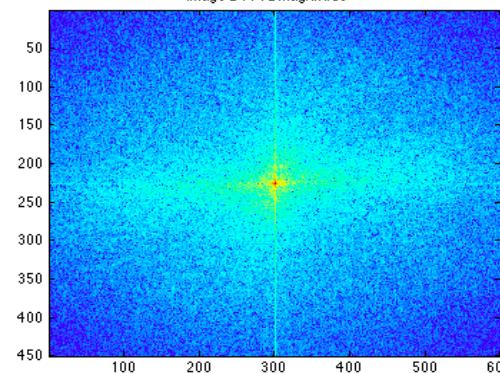
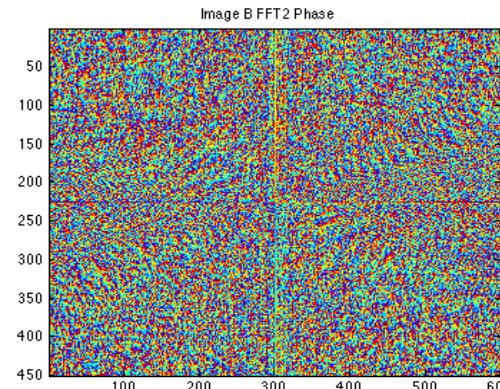
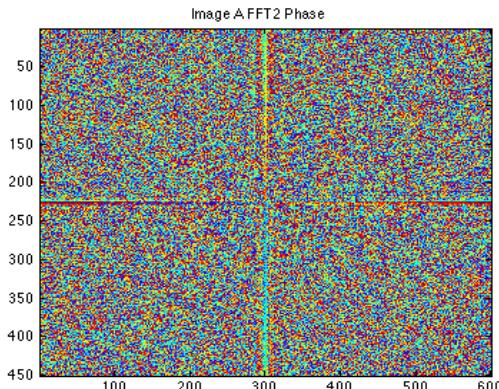


Image B FFT2 Magnitude



Phase



# CODING TIME: Magnitude and Phase in Fourier

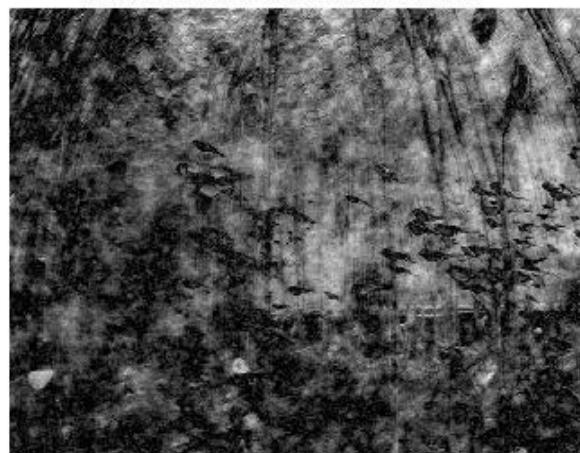
Image A



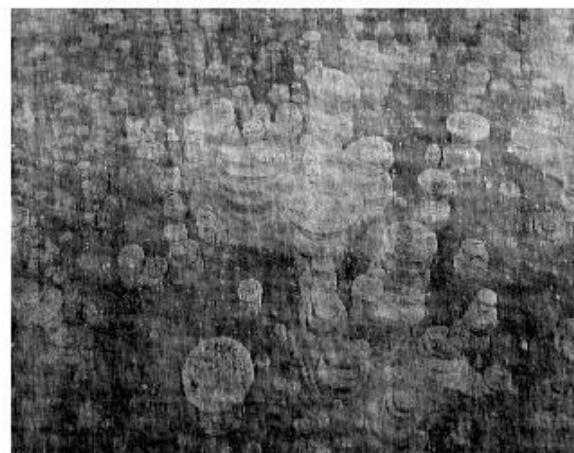
Image B



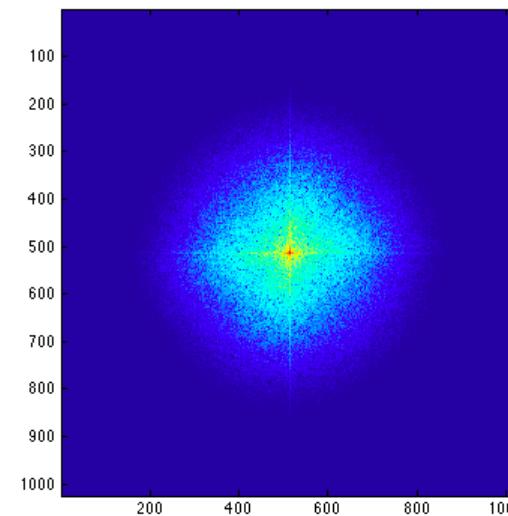
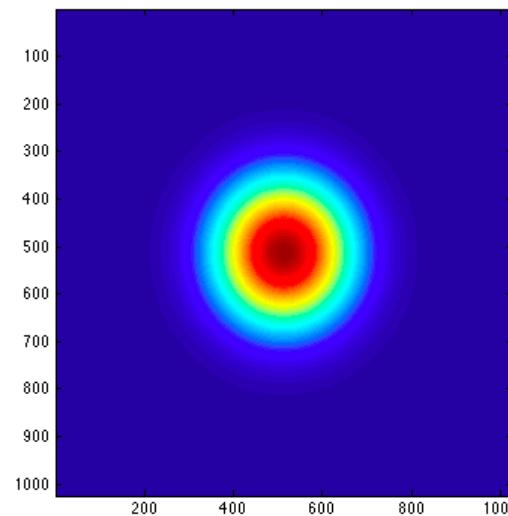
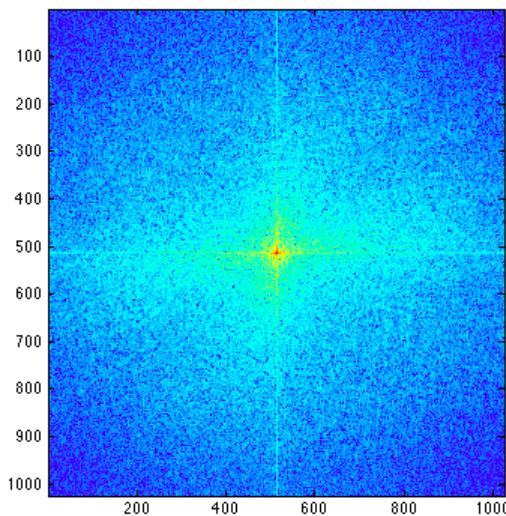
Magnitude from Image A and Phase from Image B



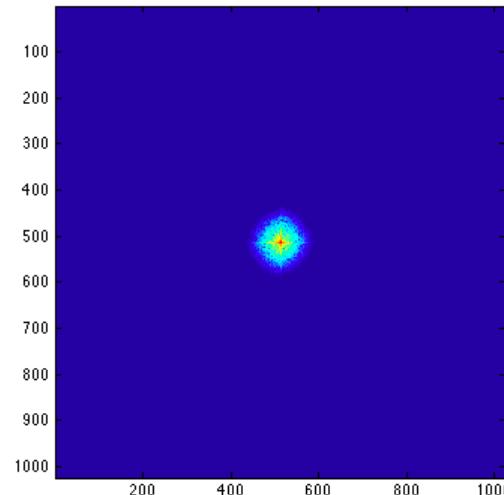
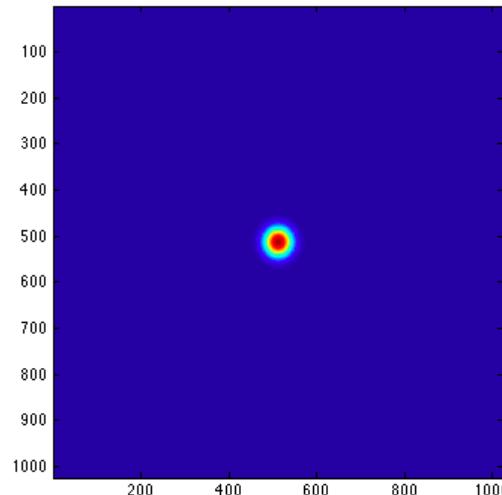
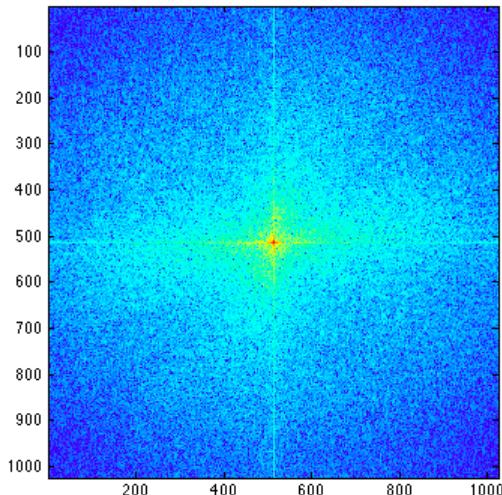
Magnitude from Image B and Phase from Image A



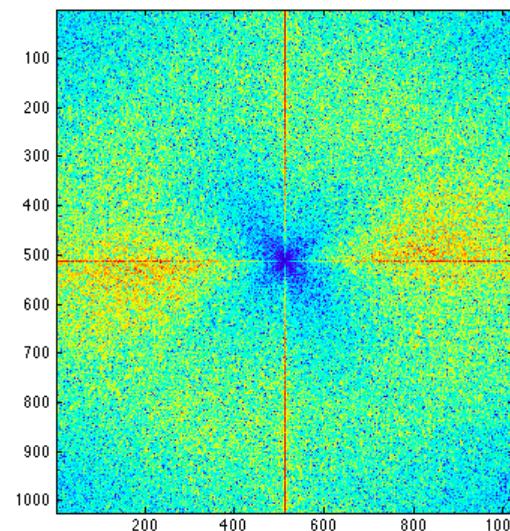
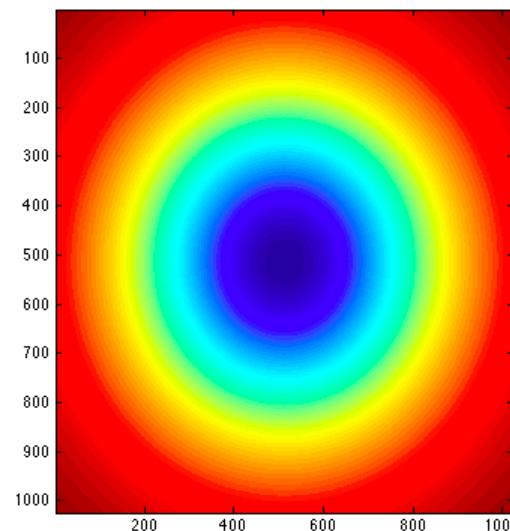
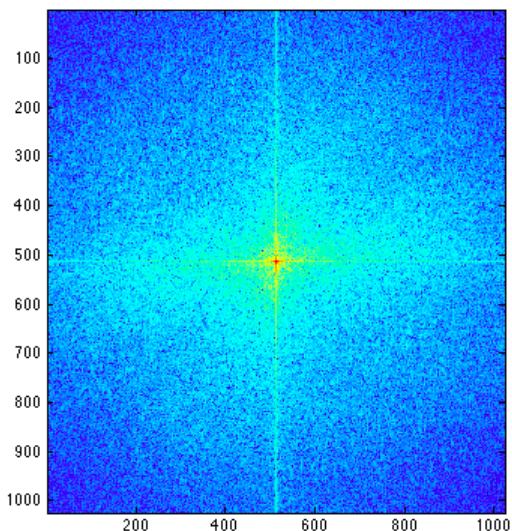
# CODING TIME: Basic filtering in Fourier



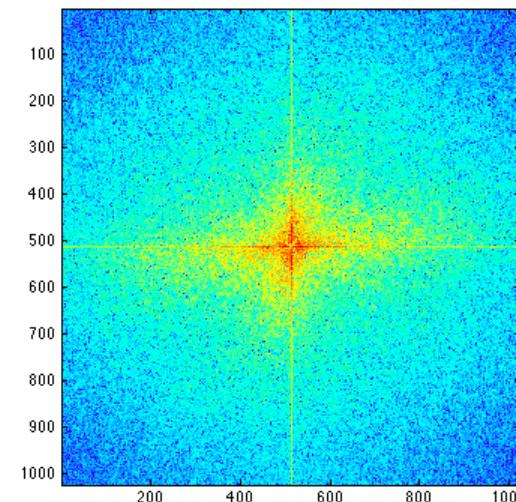
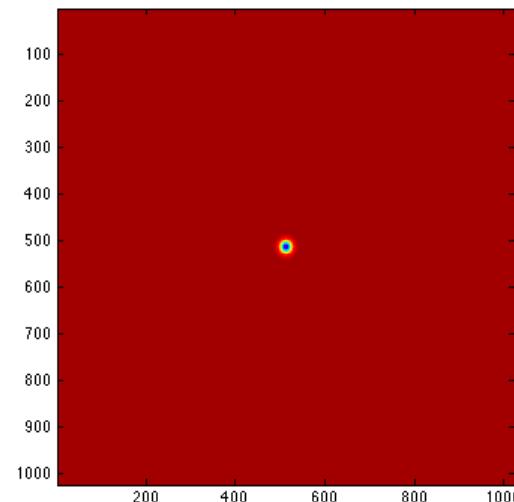
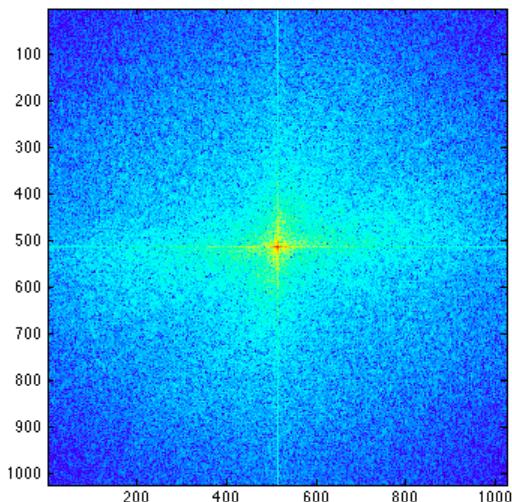
# CODING TIME: Basic filtering in Fourier



# CODING TIME: Basic filtering in Fourier



# CODING TIME: Basic filtering in Fourier

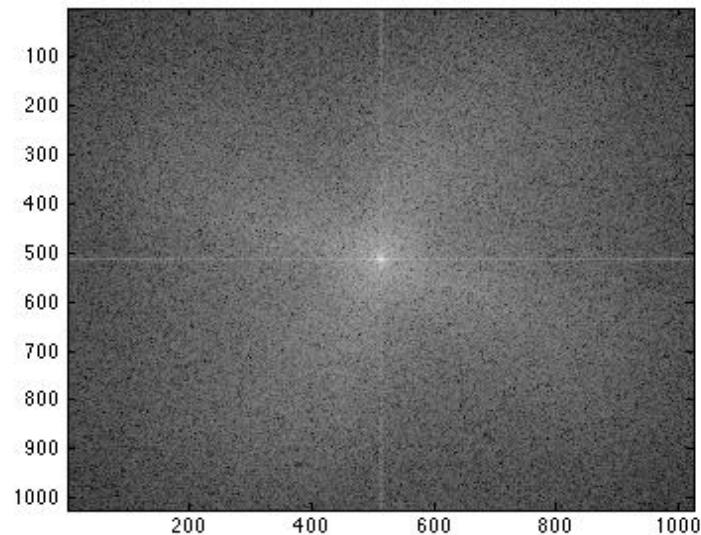


# CODING TIME: Fourier transform in 2D

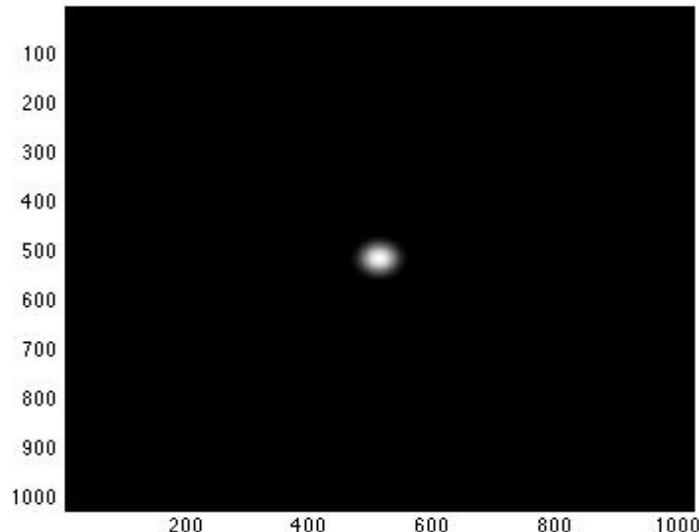
original



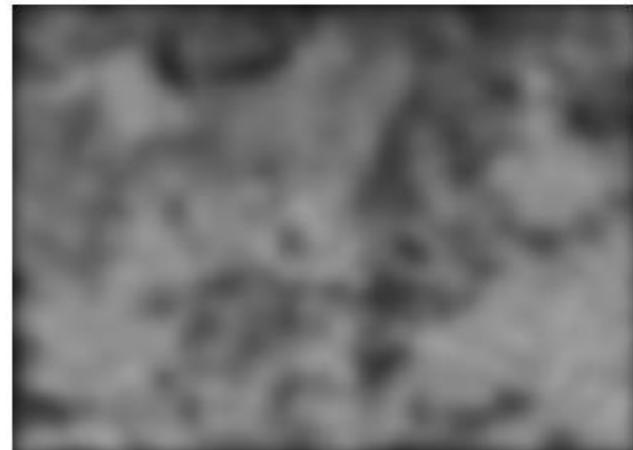
Fourier transform



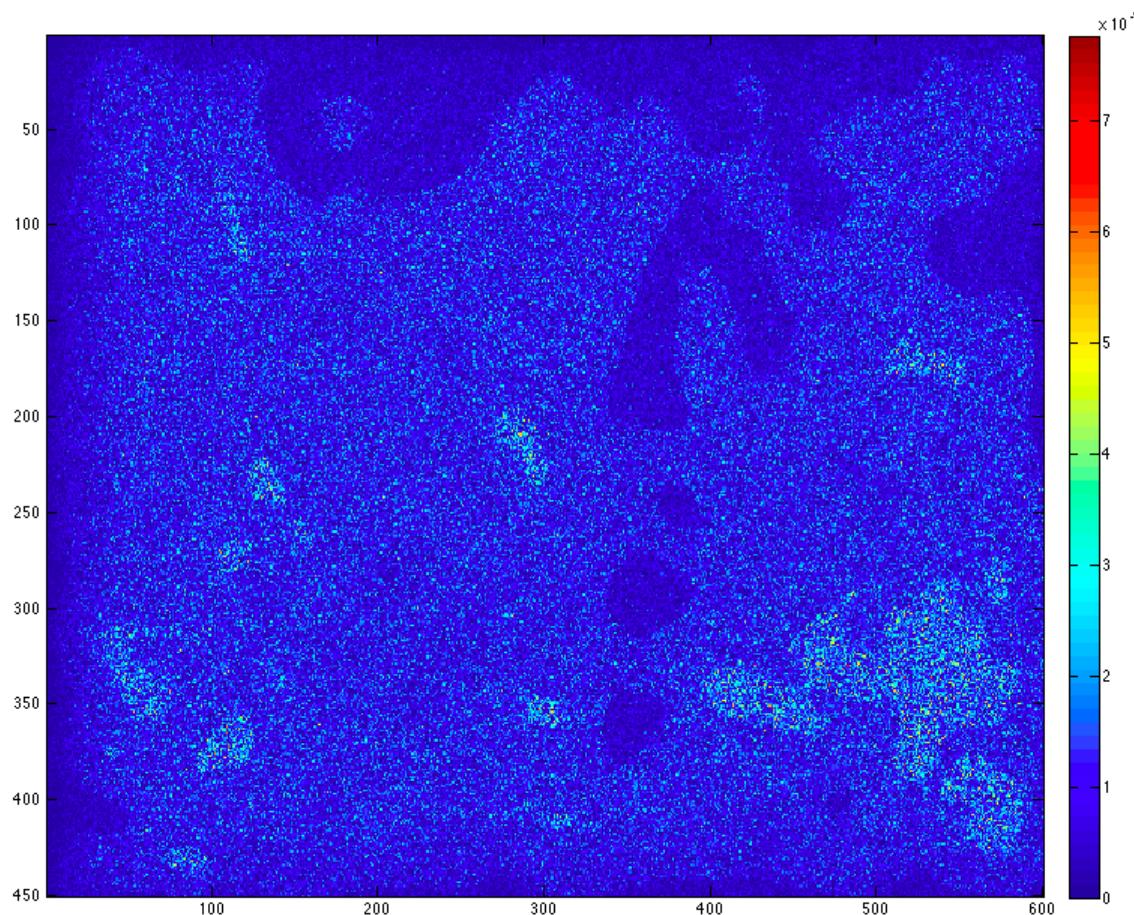
Fourier transform of the filter



result



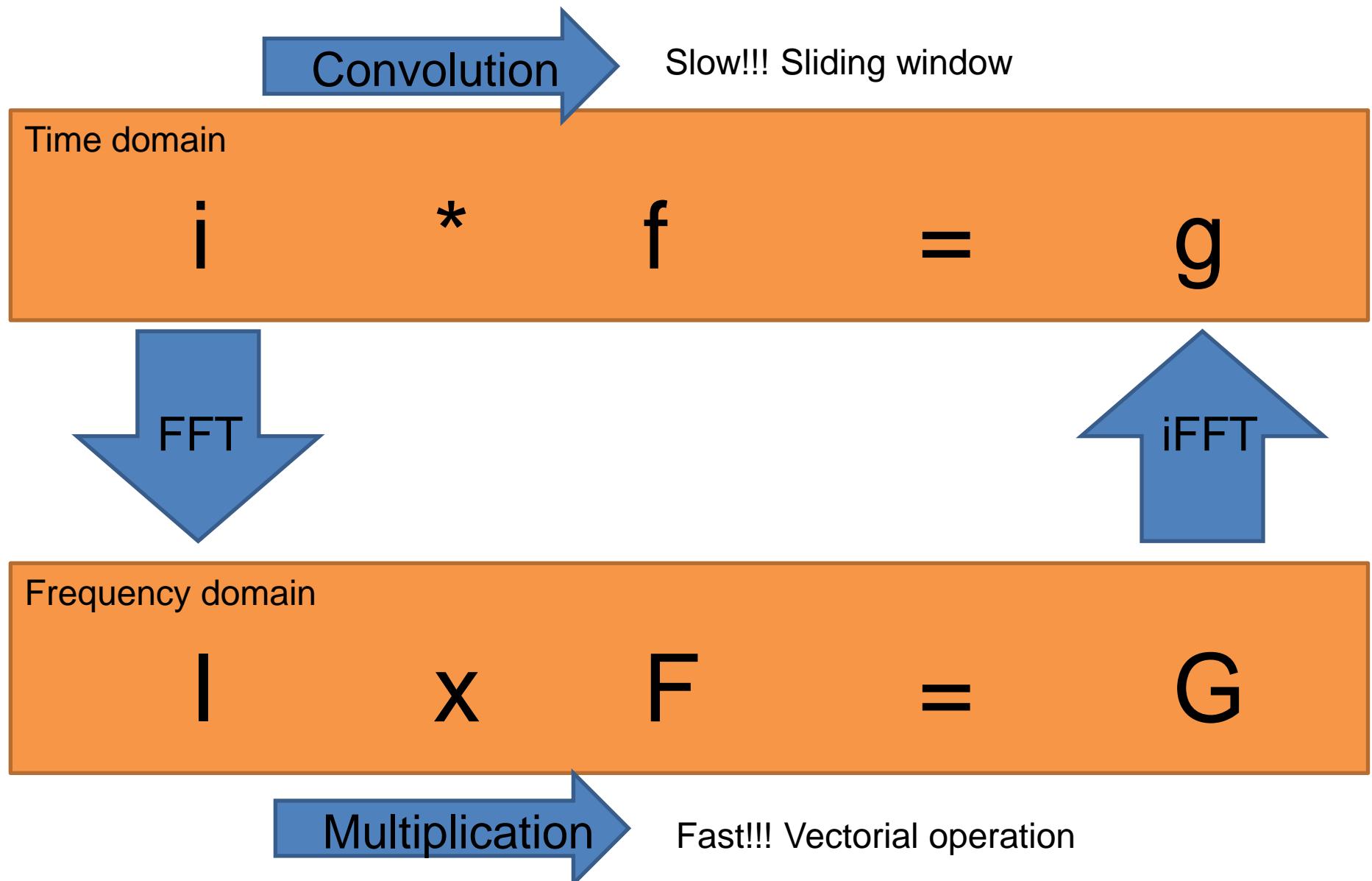
# CODING TIME: Absolute difference between convolution and the Fourier counterpart



# CODING TIME: 5 steps for using Fourier transform

```
Disp('(1) Transform the original image. Fftsize should be order  
of 2 and include padding')  
fftsize = 1024;  
im_fft = fft2(im, fftsize, fftsize);  
  
disp('We create a gaussian filter of half size value of 50')  
hs = 50;  
fil = fspecial('gaussian', hs*2+1, 10);  
  
disp('(2) fft filter and pad to the same size as the image')  
fil_fft = fft2(fil, fftsize, fftsize);  
  
disp('(3) multiply fft images')  
im_fil_fft = im_fft .* fil_fft;  
  
disp('(4) inverse fft2')  
im_fil = ifft2(im_fil_fft);  
  
disp('(5) remove padding');  
im_fil = im_fil(1+hs:size(im,1)+hs, 1+hs:size(im, 2)+hs);
```

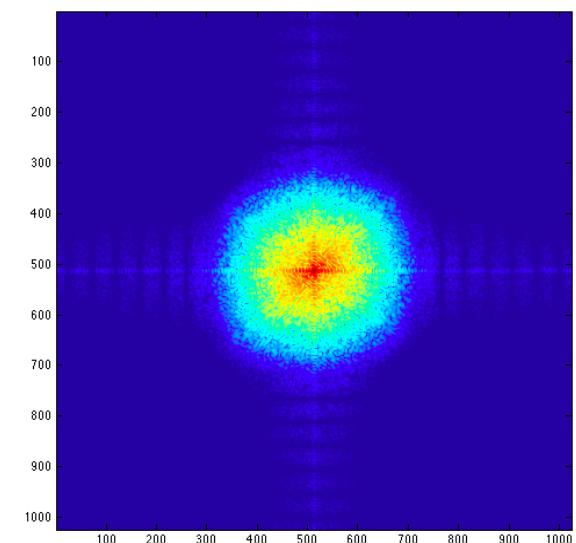
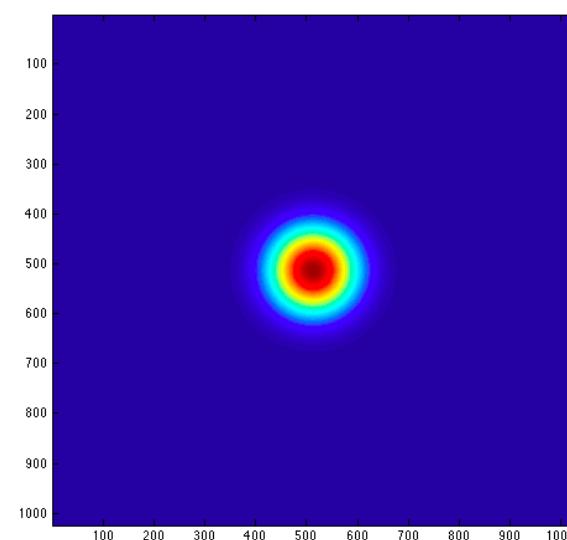
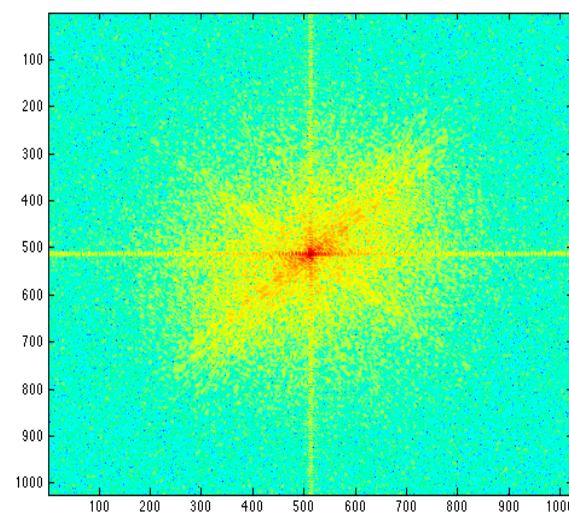
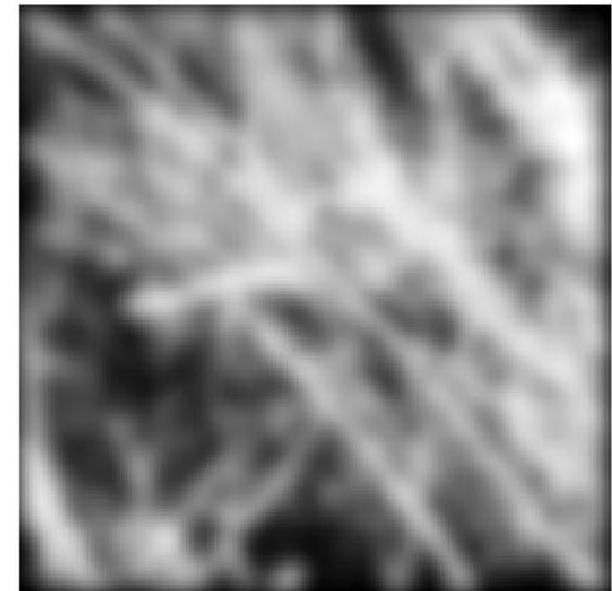
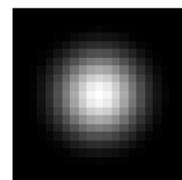
# Convolution using Fourier



# Convolution using Fourier

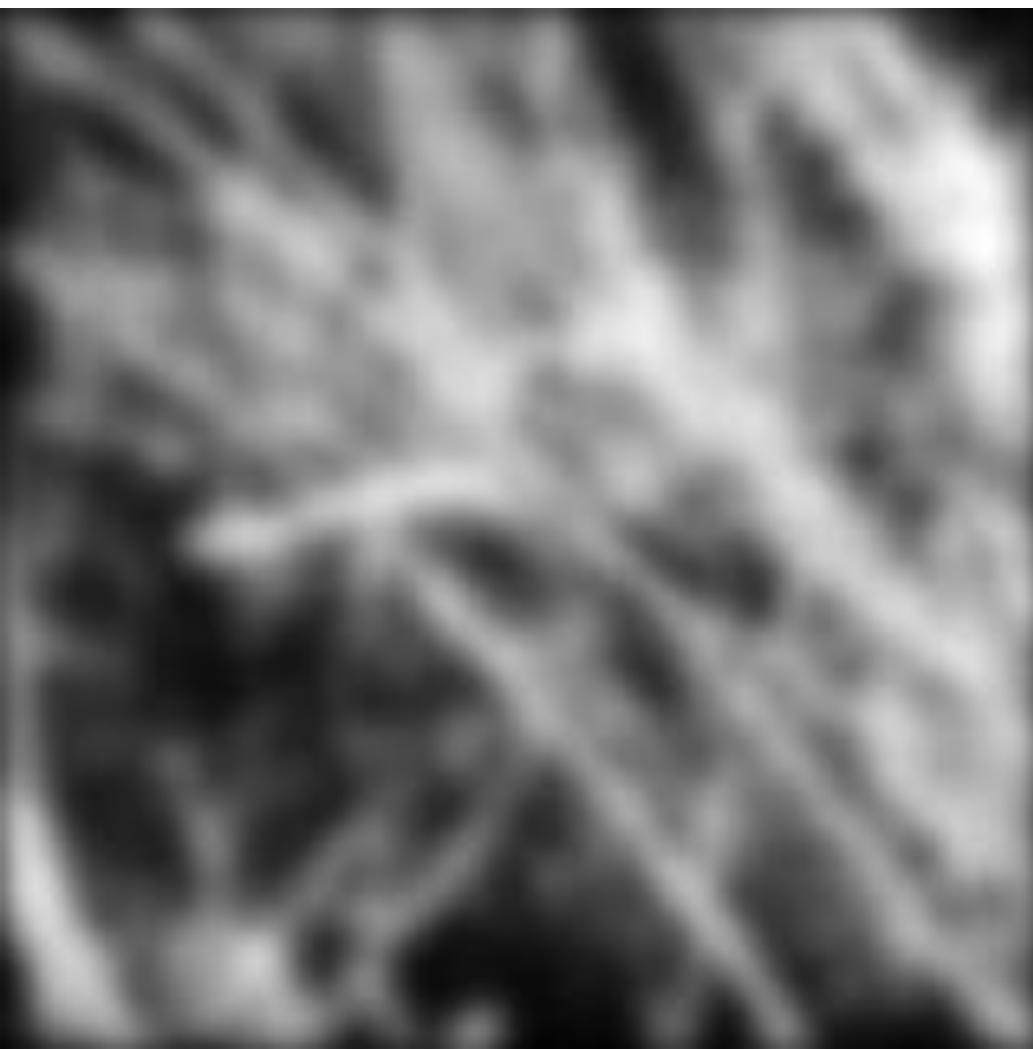


Gaussian Filter

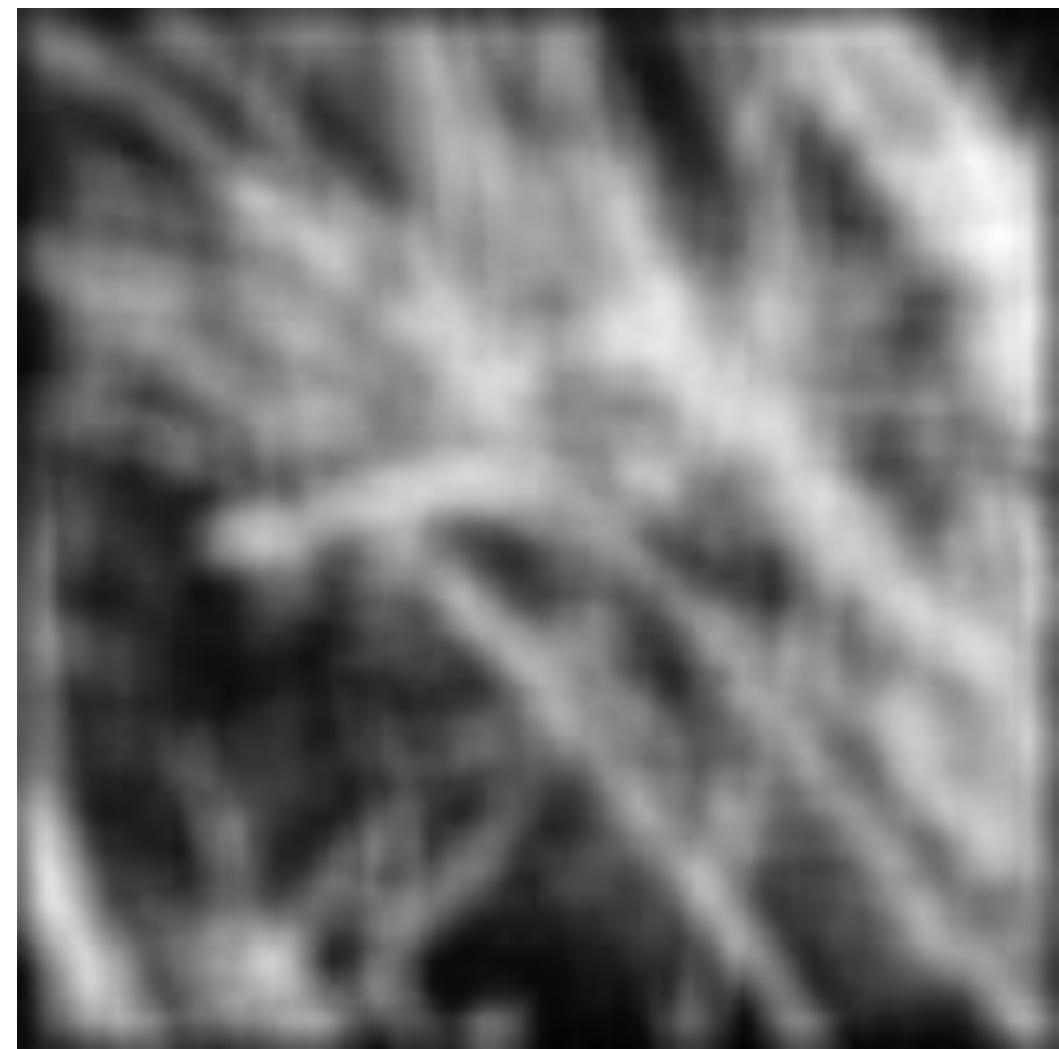


# WHY DOES BOX FILTER SMOOTHING HAVE SO MANY ARTIFACTS?

Gaussian



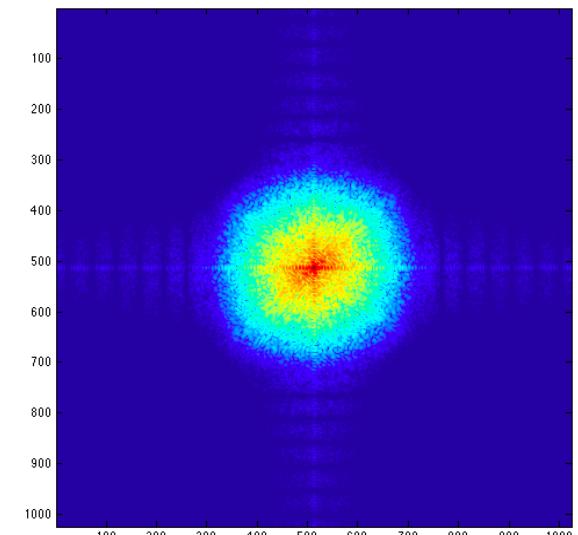
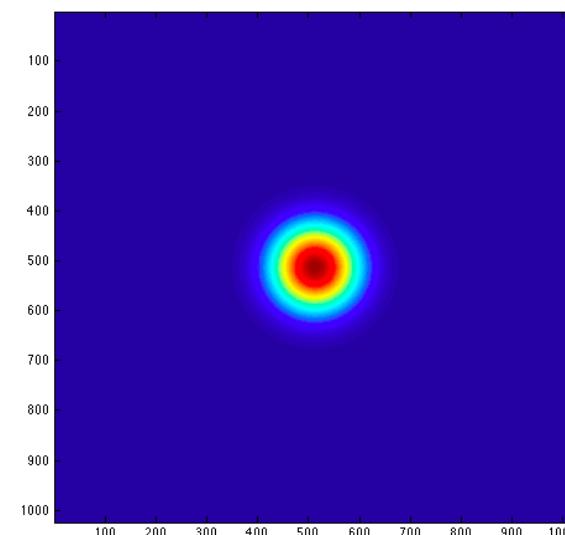
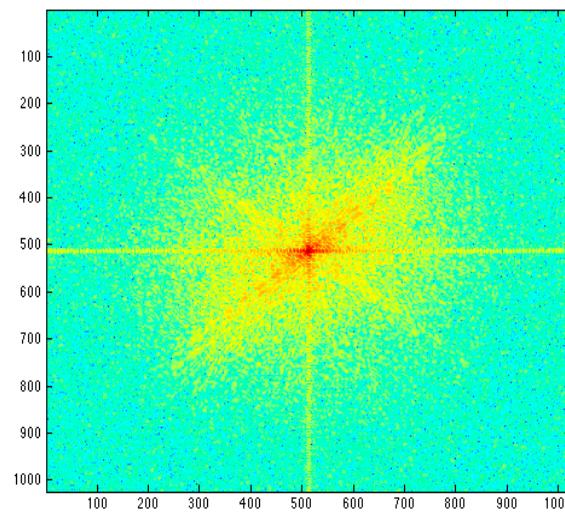
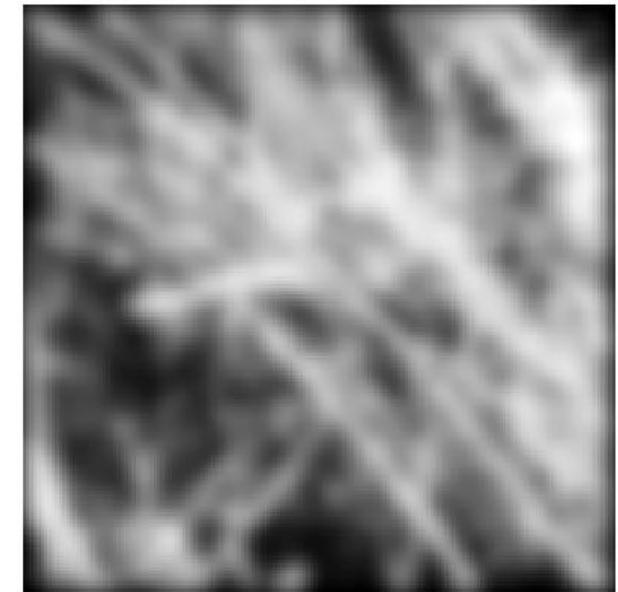
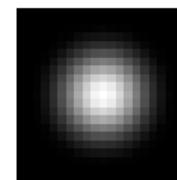
Box filter



# WHY DOES BOX FILTER SMOOTHING HAVE SO MANY ARTIFACTS?



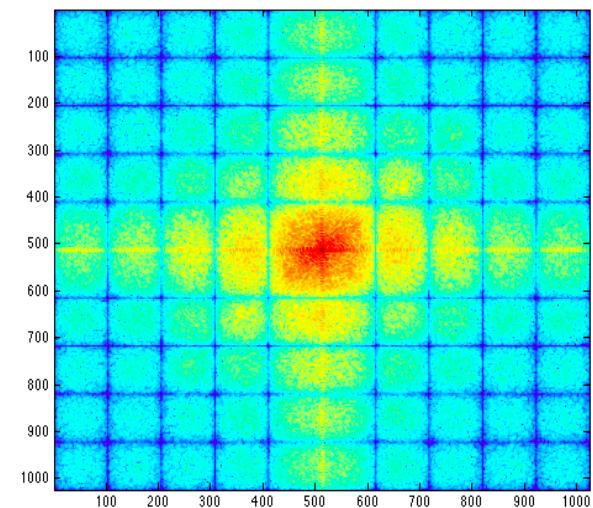
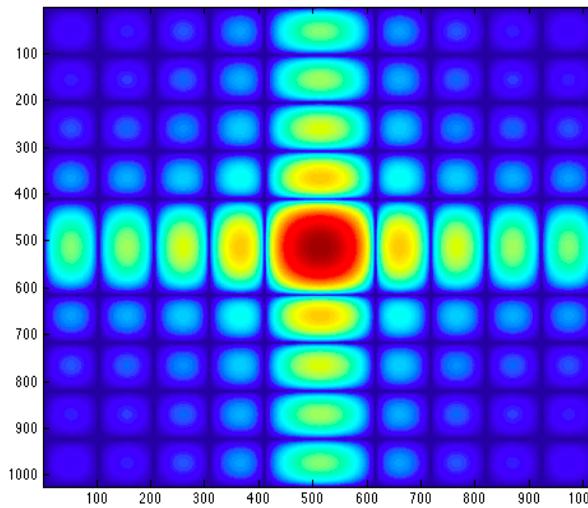
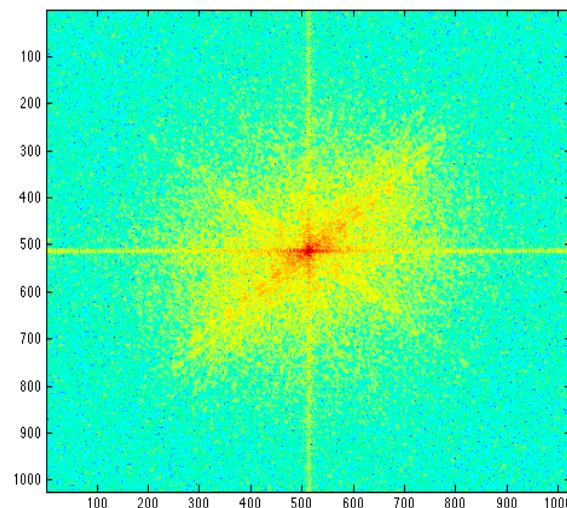
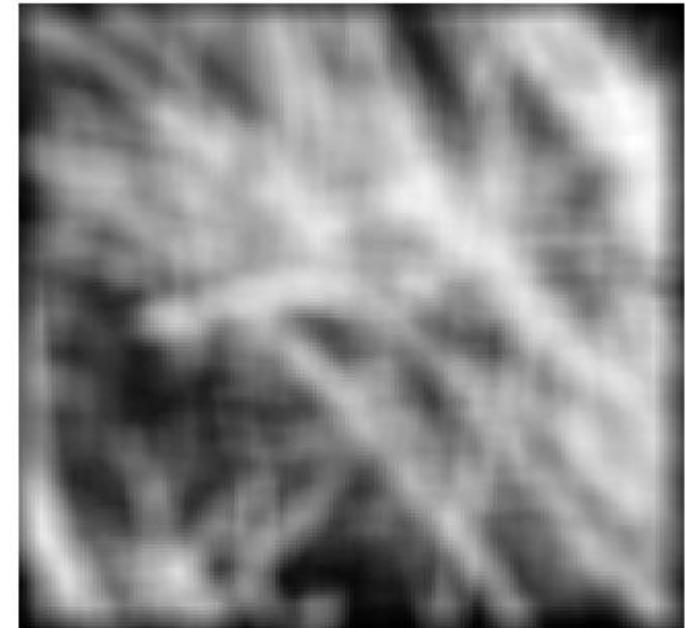
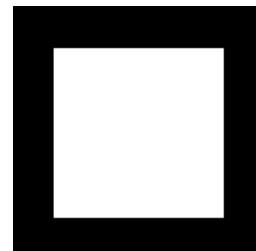
Gaussian Filter



# WHY DOES BOX FILTER SMOOTHING HAVE SO MANY ARTIFACTS?

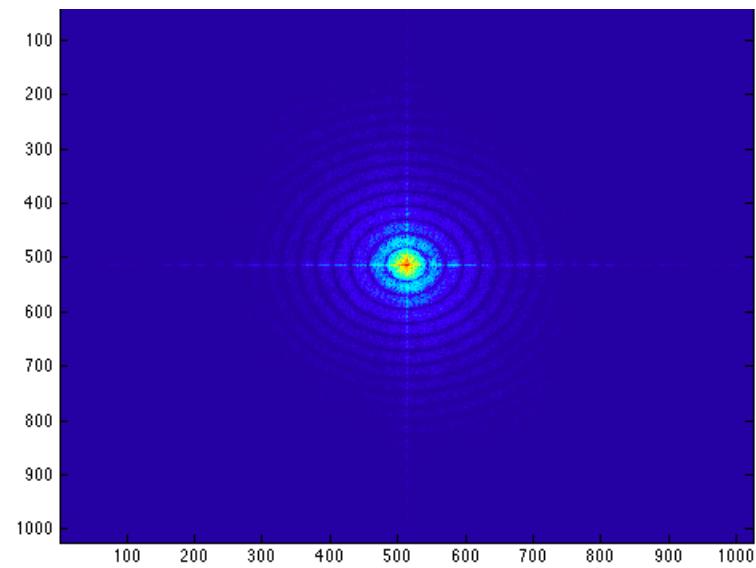


Box Filter



# Inverse filtering (for academic purposes)

$$G(w) = I(w)H(w) \implies I(w) = G(w)/H(w)$$



# CODING TIME: Inverse filtering (for academic purposes)

$$G(w) = I(w)H(w) \implies I(w) = G(w)/H(w)$$

```
psf=fspecial('disk',20);
filtered=conv2(im,psf);

fft_filtered=fft2(filtered,1024,1024);
fft_psf=fft2(psf,1024,1024);
fft_recons=fft_filtered./fft_psf;

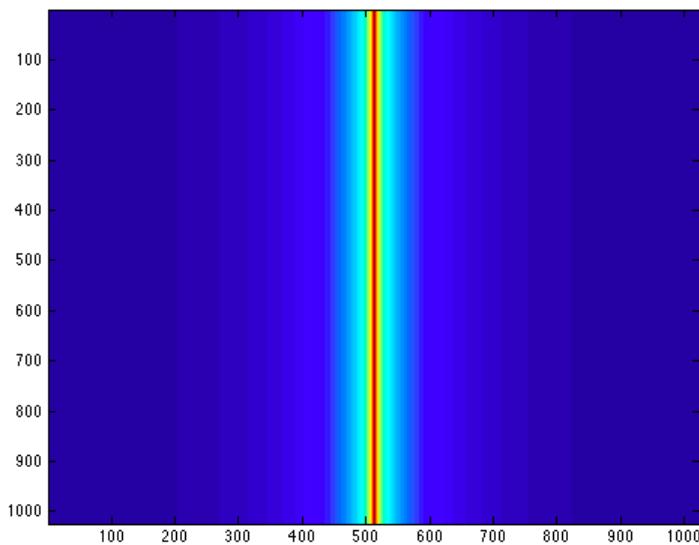
recons=ifft2(fft_recons);
```

We must know the degradation model  $H(w)$   
(aka **Point Spread Function PSF**).

# CODING TIME: Inverse filtering (for academic purposes)



input



output

# CODING TIME: Inverse filtering and borders

input



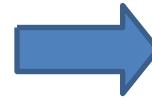
# CODING TIME: Inverse filtering and borders

output

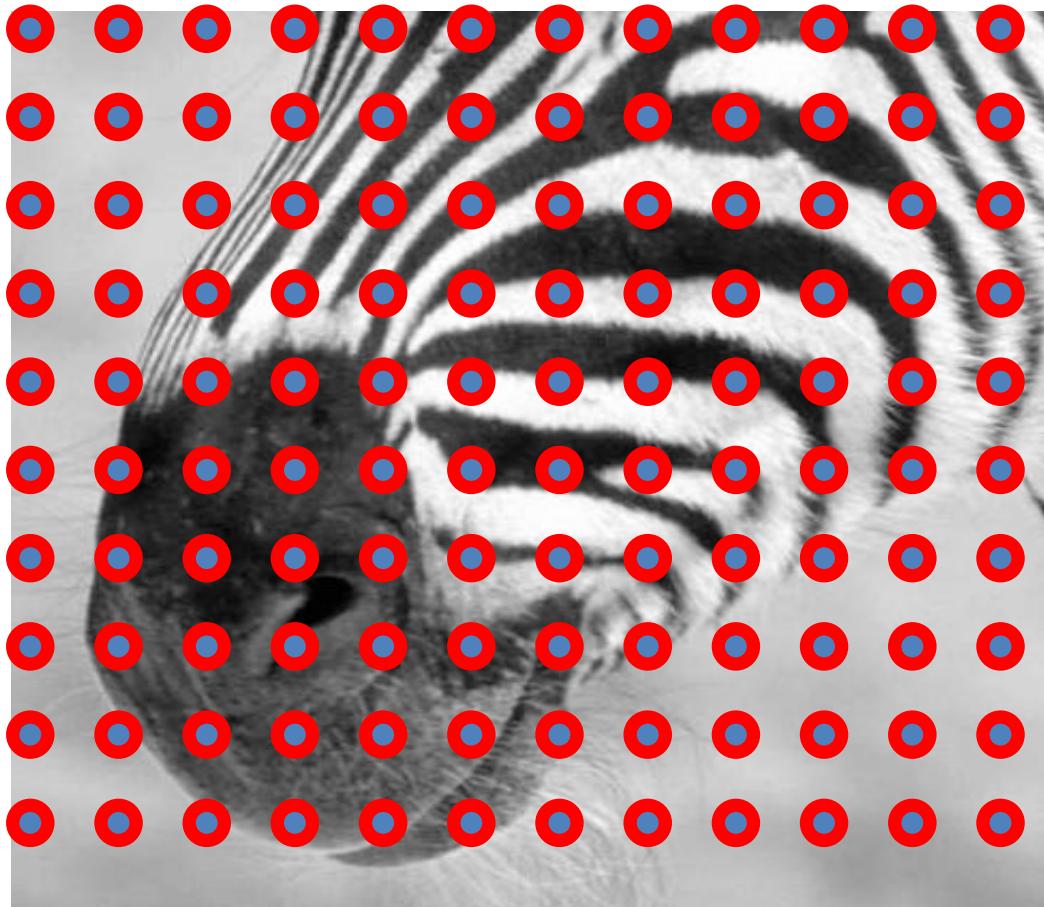


# Sampling

# WHY DOES A LOWER RESOLUTION IMAGE STILL MAKE SENSE TO US? WHAT DO WE LOSE?



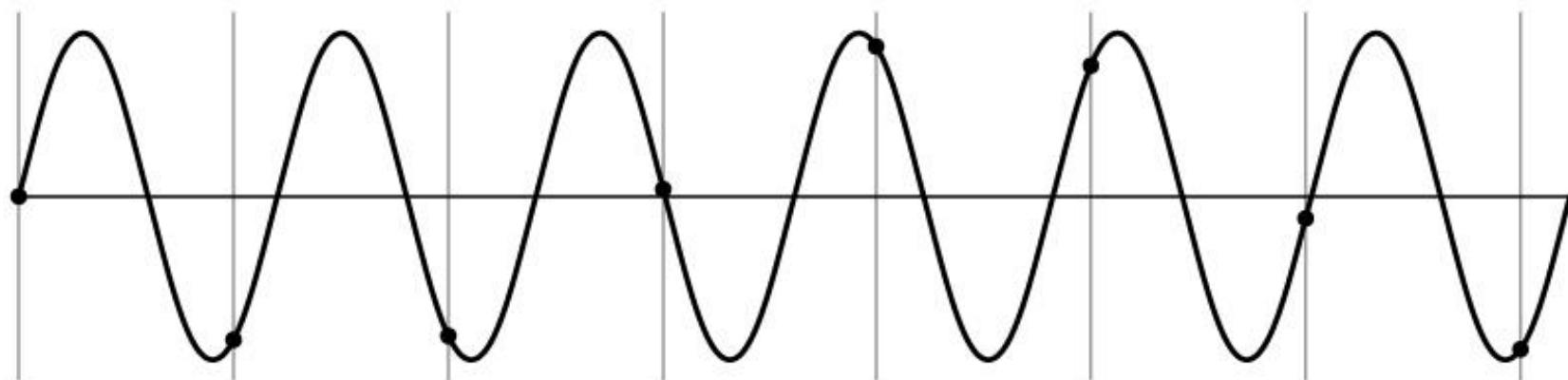
# SUBSAMPLING BY A FACTOR OF 2



Throw away every other row and column  
to create a 1/2 size image

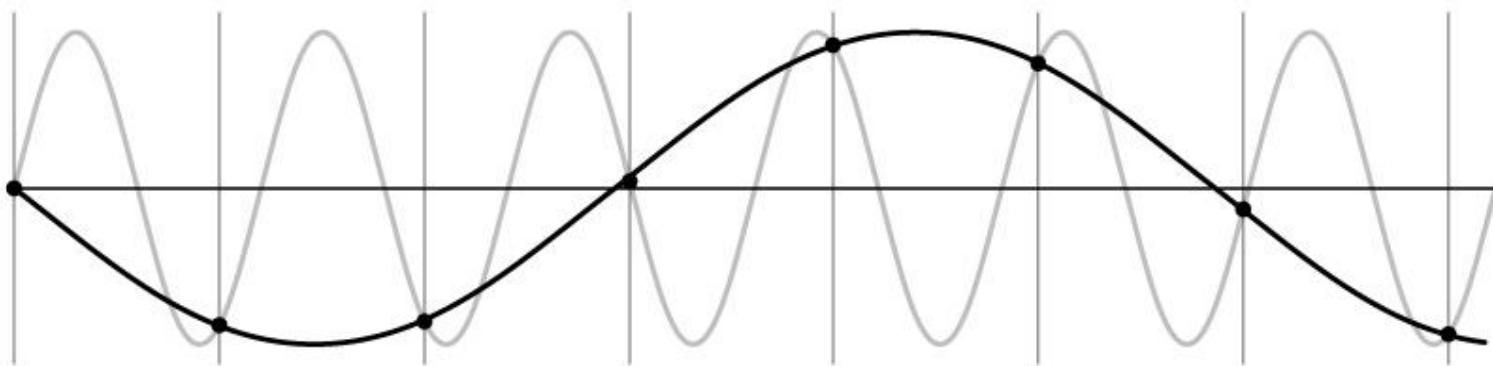
# ALIASING PROBLEM

- 1D example (sinewave):



# ALIASING PROBLEM

- 1D example (sinewave):

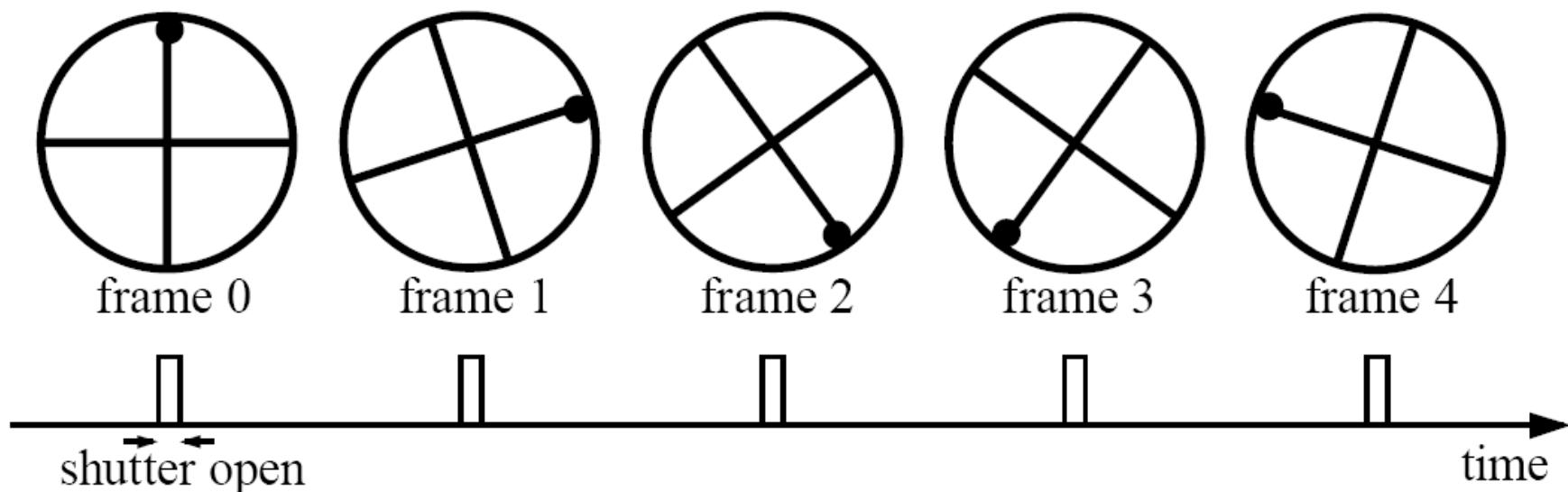


# ALIASING IN VIDEO

Imagine a spoked wheel moving to the right (rotating clockwise).

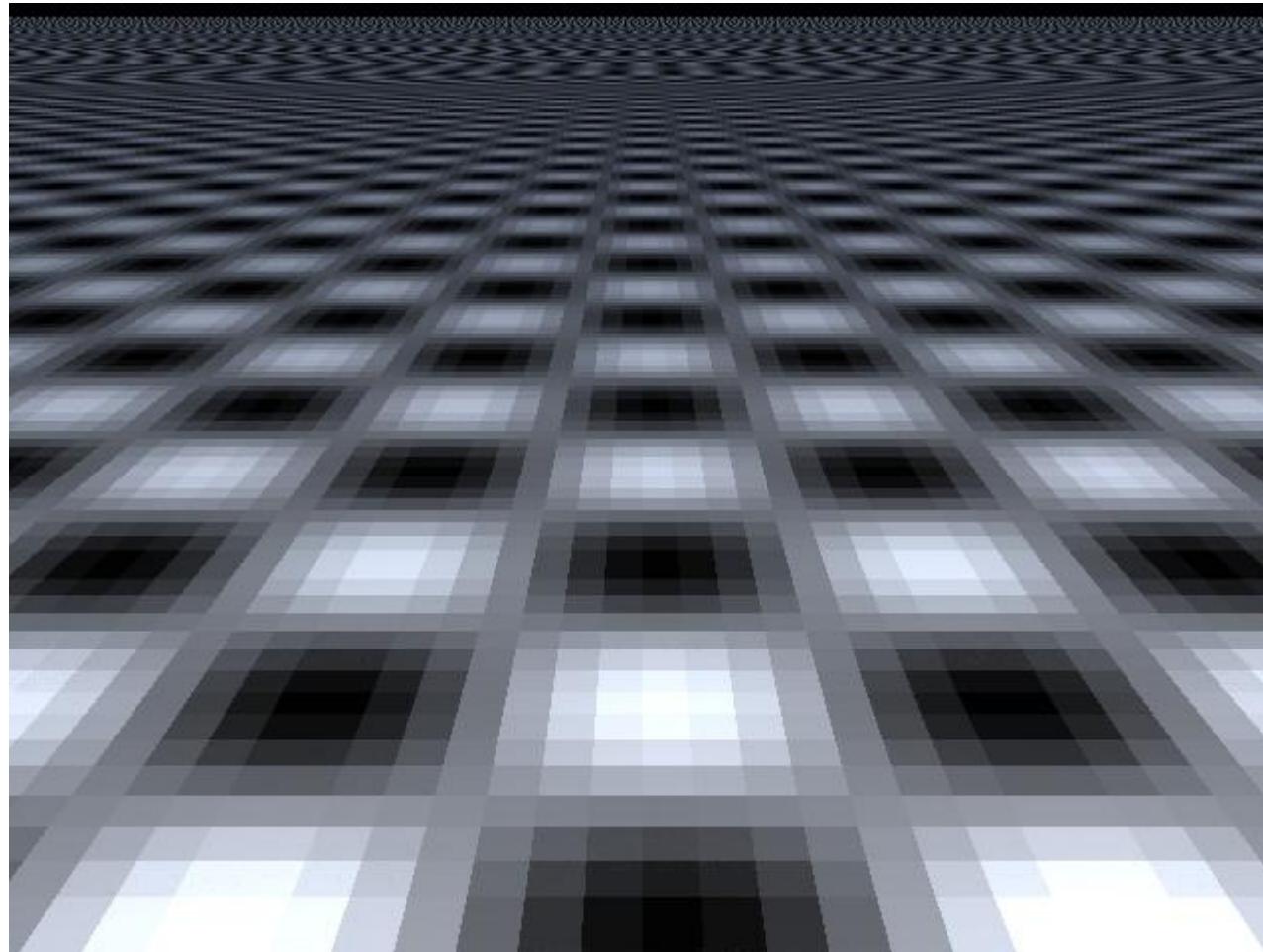
Mark wheel with dot so we can see what's happening.

If camera shutter is only open for a fraction of a frame time (frame time =  $1/30$  sec. for video,  $1/24$  sec. for film):



Without dot, wheel appears to be rotating slowly backwards!  
(counterclockwise)

# ALIASING IN CG



# SAMPLING AND ALIASING

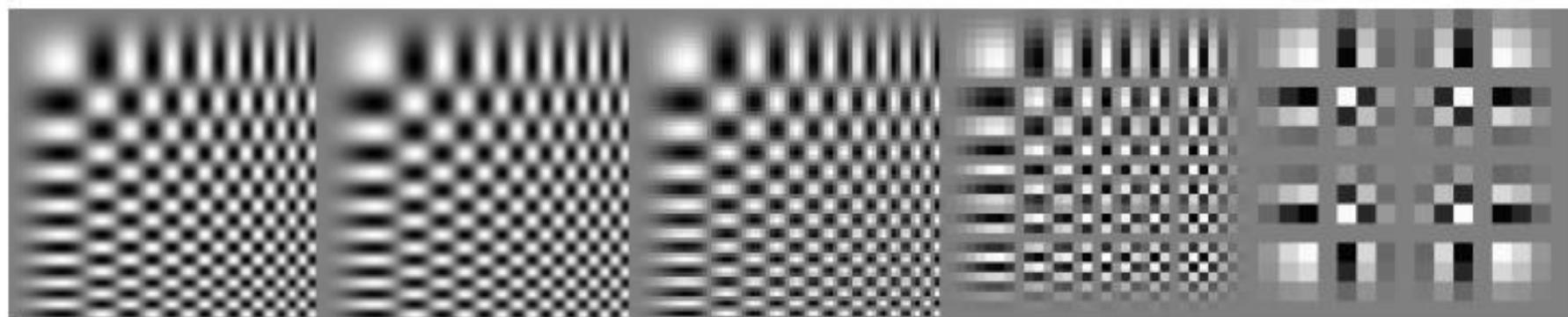
256x256

128x128

64x64

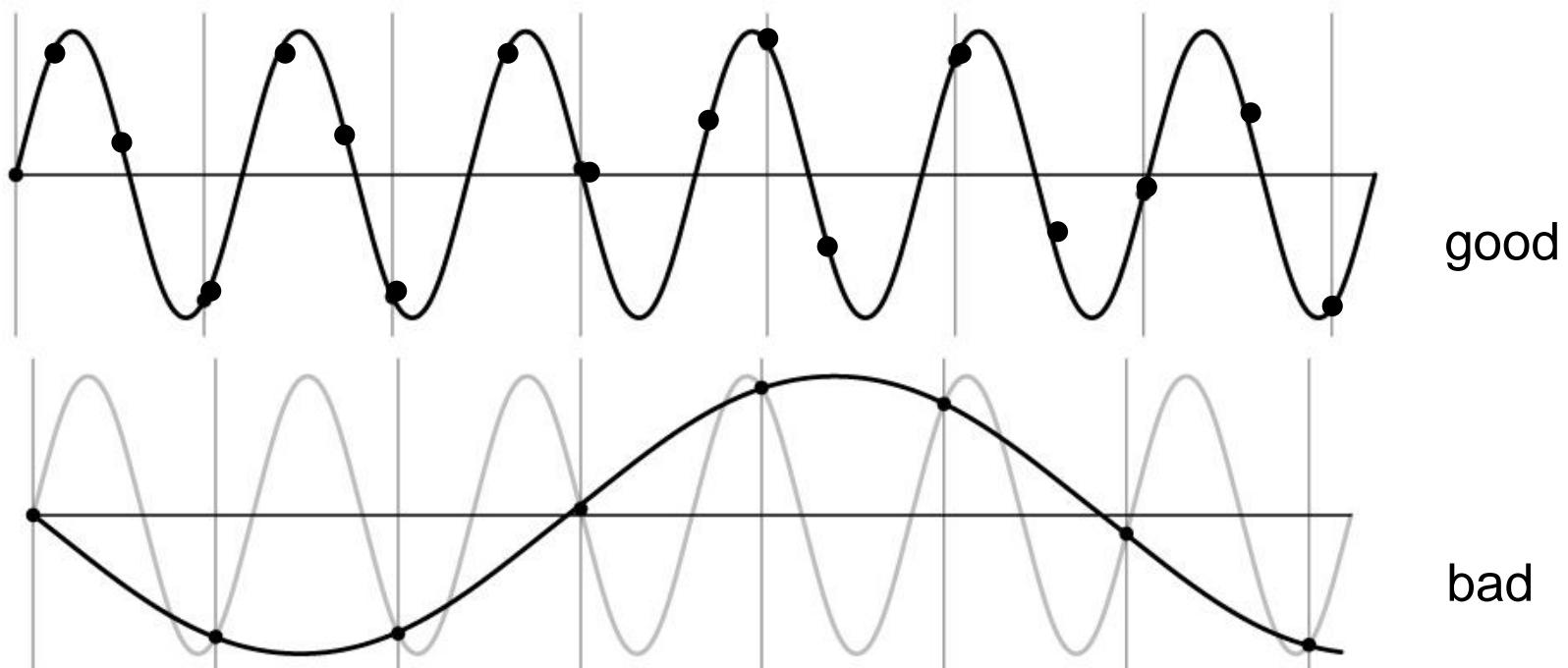
32x32

16x16



# NYQUIST-SHANNON SAMPLING THEOREM

- When sampling a signal at discrete intervals, the sampling frequency must be  $f_{sampling} \geq 2f_{max}$
- 
- $f_{max}$  = max frequency of the input signal
- This will allow to reconstruct the original perfectly from the sampled version



# ANTI-ALIASING

Solutions:

- Sample more often
- Get rid of all frequencies that are greater than half the new sampling frequency
  - Will lose information
  - But it's better than aliasing
  - Apply a smoothing filter

# DOWNSAMPLING ALGORITHM

1. Start with  $\text{image}(h, w)$

2. Apply low-pass filter

```
im.blur = imfilter(image, fspecial('gaussian', 7, 1))
```

1. Sample every other pixel

```
im.small = im.blur(1:2:end, 1:2:end);
```

# ANTI-ALIASING

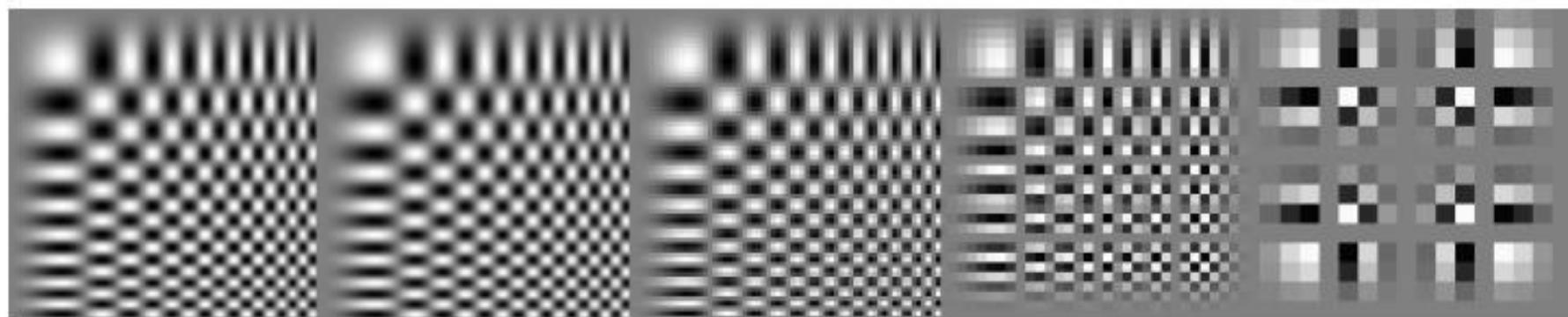
256x256

128x128

64x64

32x32

16x16



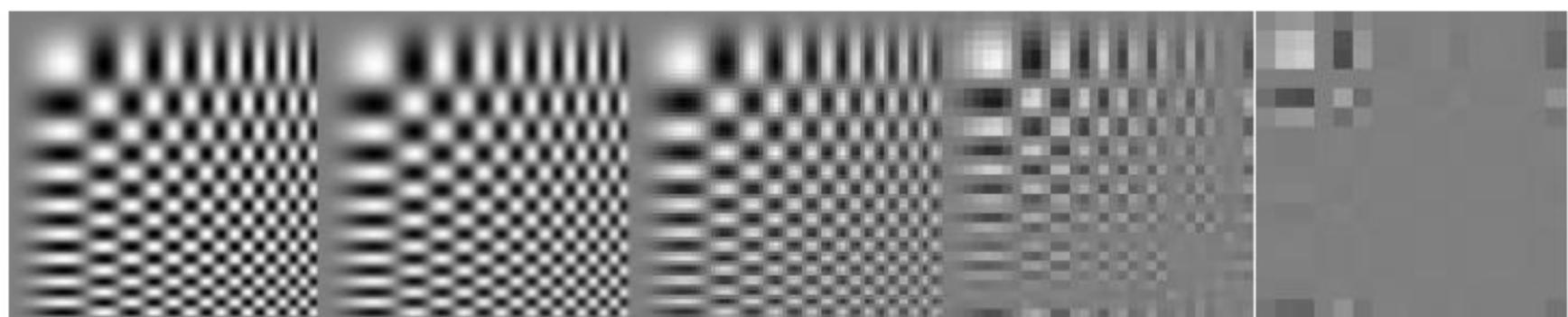
256x256

128x128

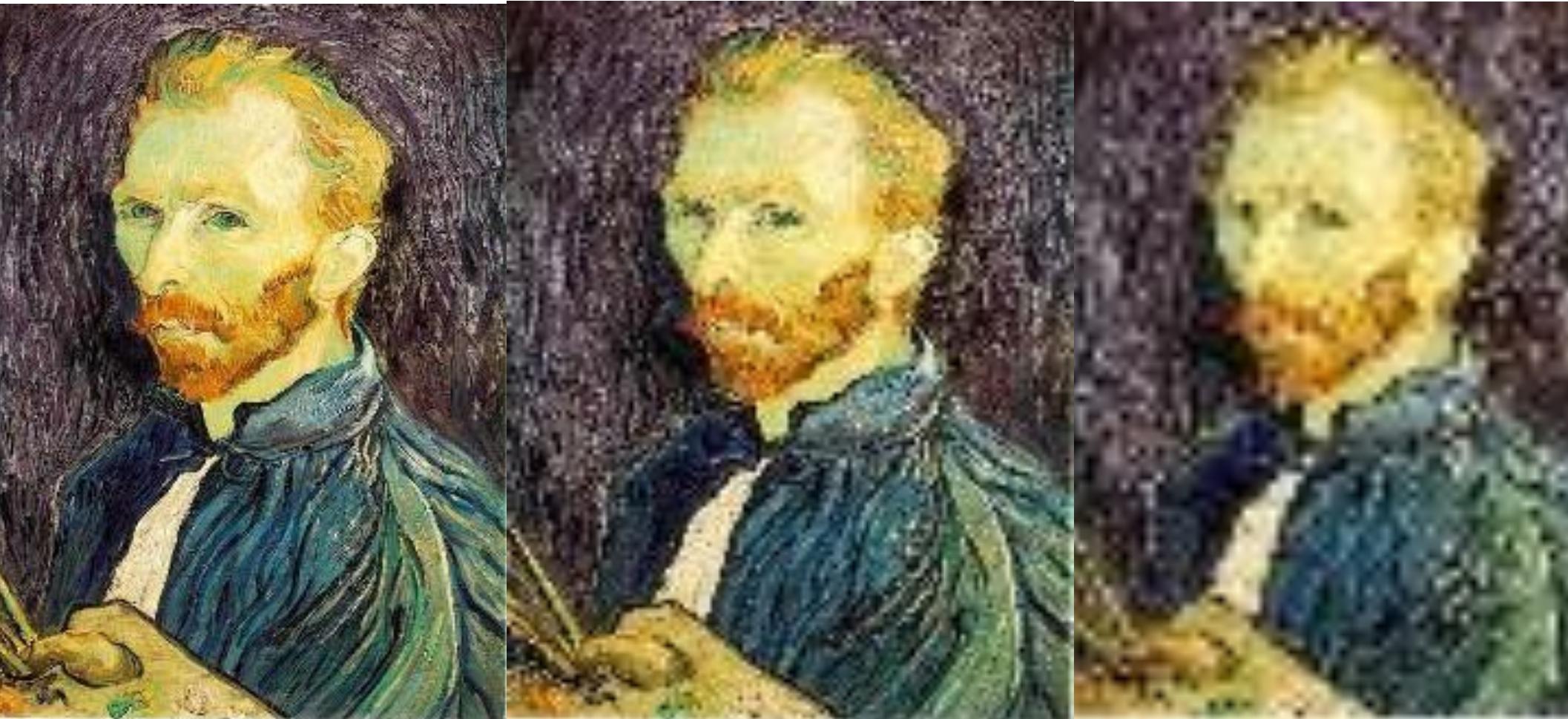
64x64

32x32

16x16



# SUBSAMPLING WITHOUT PREFILTERING



1/2

1/4 (2x zoom)

1/8 (4x zoom)

# SUBSAMPLING WITH PREFILTERING



Gaussian 1/2

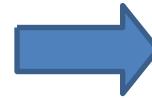


G 1/4

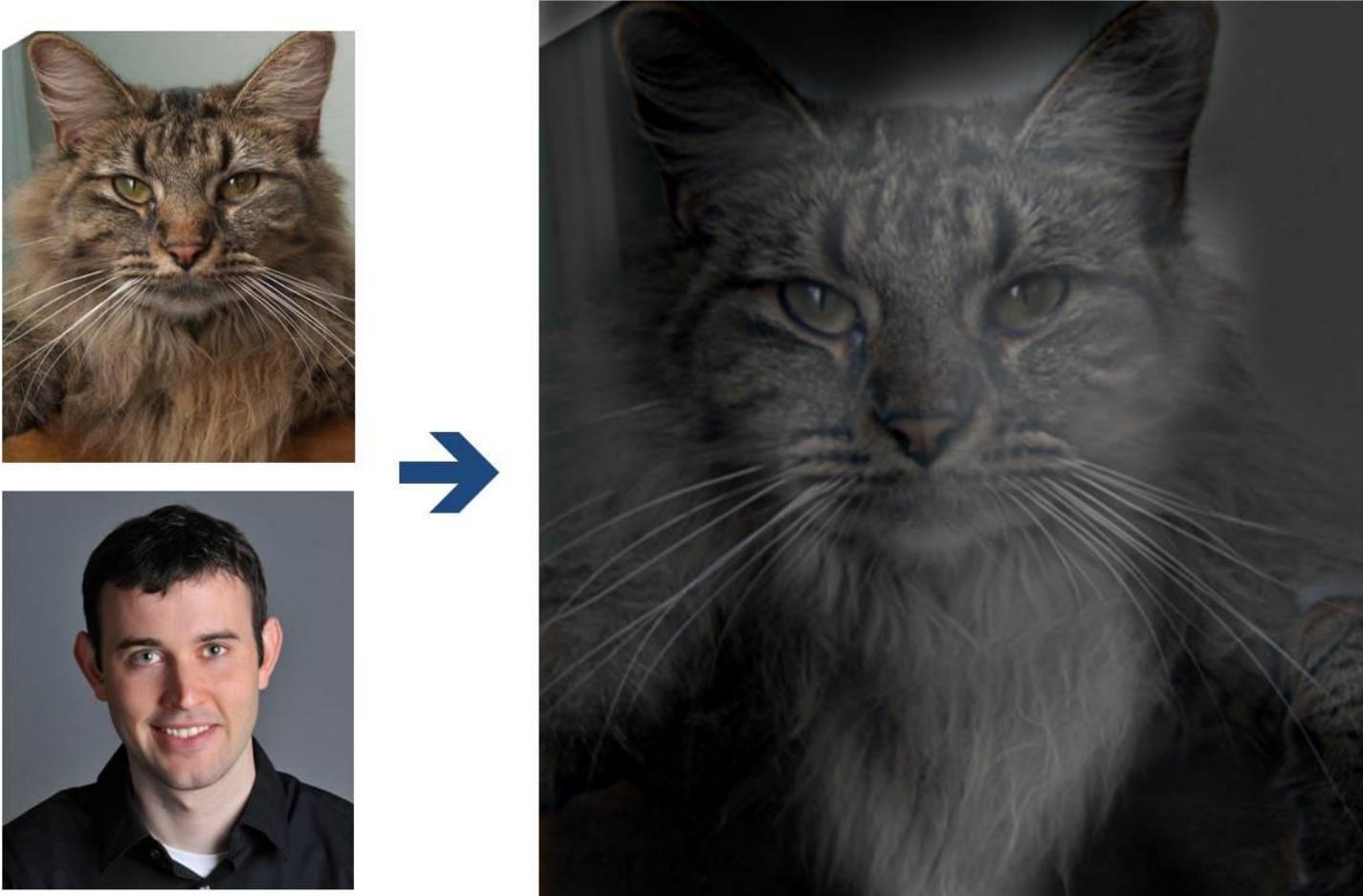


G 1/8

# WHY DOES A LOWER RESOLUTION IMAGE STILL MAKE SENSE TO US? WHAT DO WE LOSE?

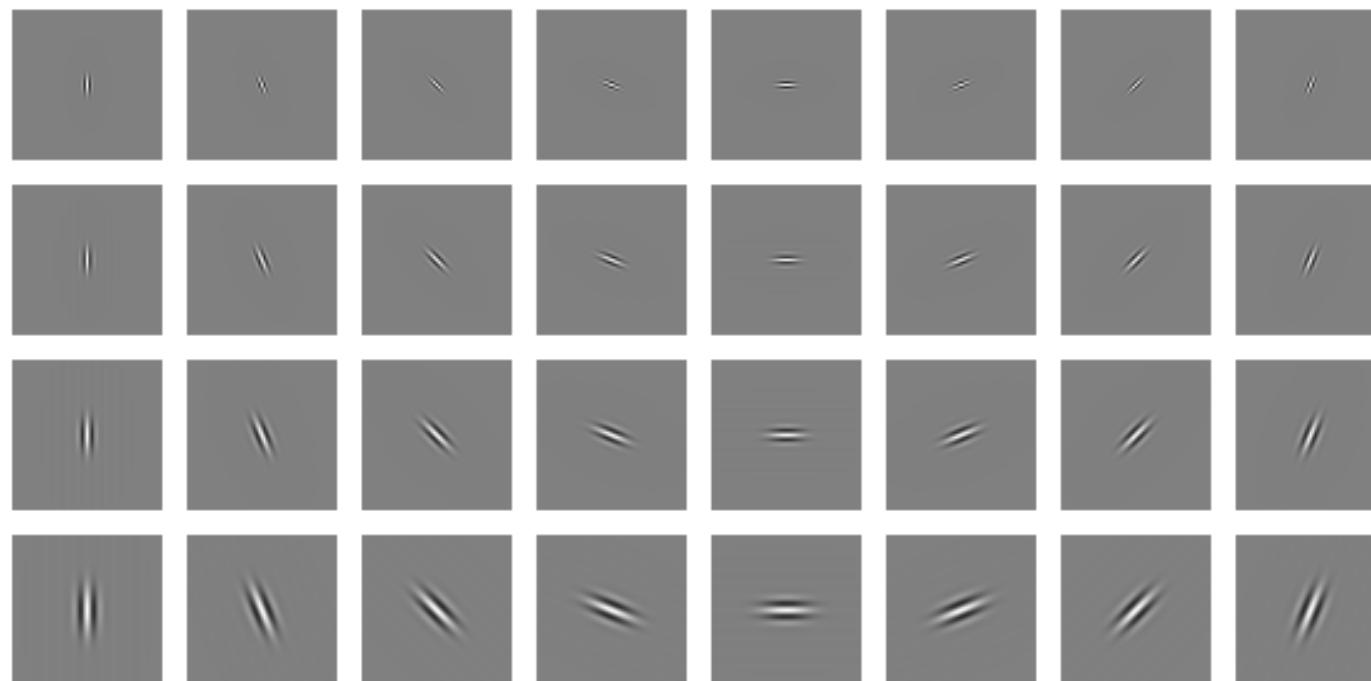


# WHY DO WE GET DIFFERENT, DISTANCE-DEPENDENT INTERPRETATIONS OF HYBRID IMAGES?



# CLUES FOR HUMAN PERCEPTION

- Early processing in humans filters for various orientations and scales of frequency
- Perceptual cues in the mid frequencies dominate perception
- When we see an image from far away, we are effectively subsampling it



Early Visual Processing: Multi-scale edge and blob filters

# HIBRID IMAGE IN FFT

YOUR TURN ...

# Sharpening revisited

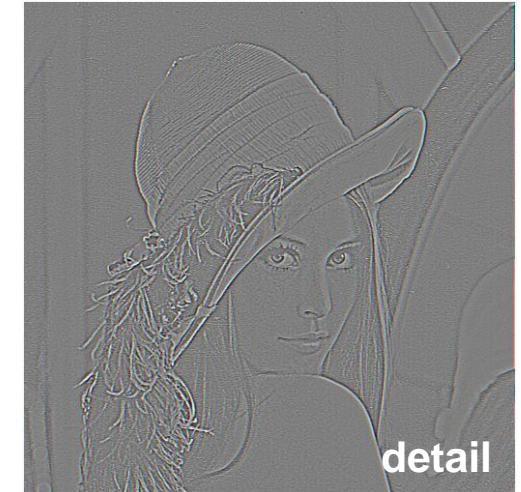
- What does blurring take away?



original



smoothed (5x5)



detail

-

=

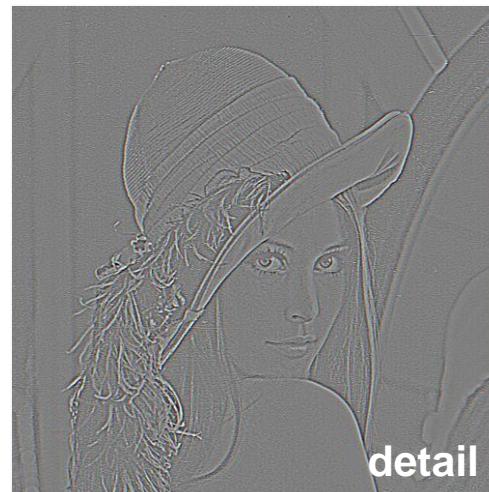
Let's add it back:



original

$+ \alpha$

=



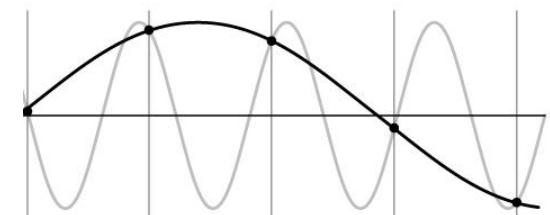
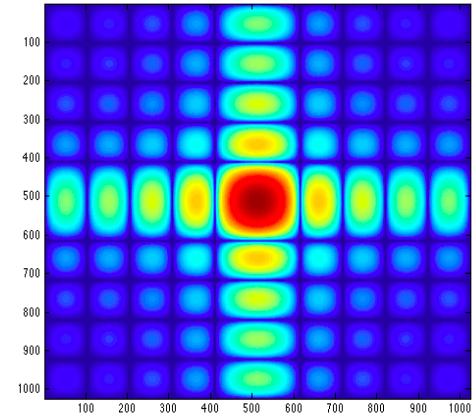
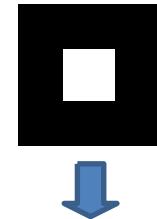
detail



sharpened

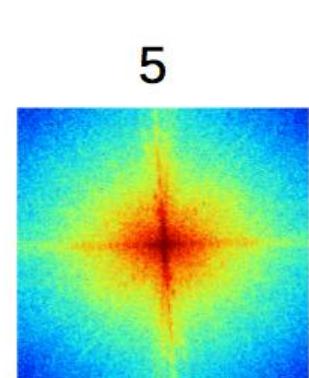
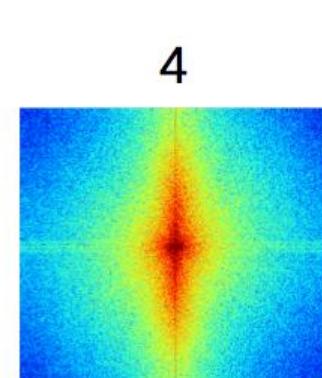
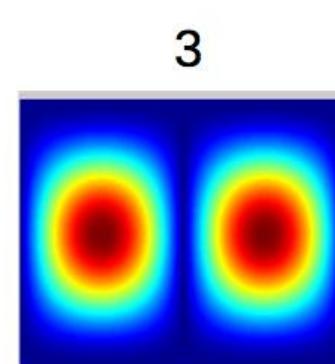
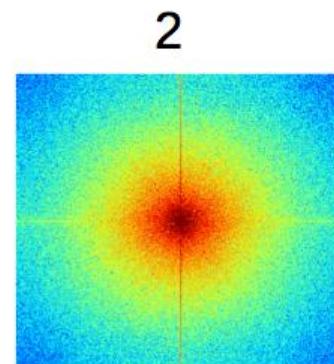
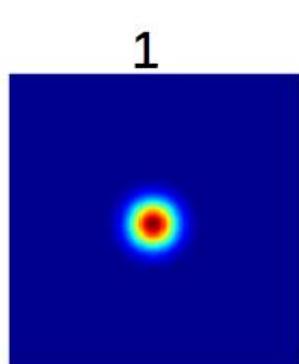
# THINGS TO REMEMBER

- Sometimes it makes sense to think of images and filtering in the frequency domain
  - Fourier analysis
- Can be faster to filter using FFT for large images ( $N \log N$  vs.  $N^2$  for auto-correlation)
- Images are mostly smooth
  - Basis for compression
- Remember to low-pass before sampling



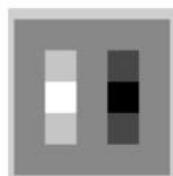
# TAKE HOME QUESTIONS

1. Match the spatial domain image to the Fourier magnitude image

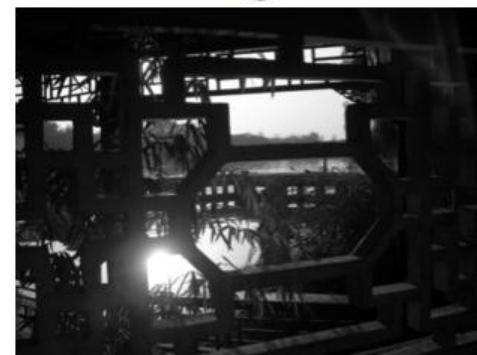


B

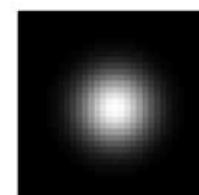
A



C



D



E



# NEXT CLASS

- Denoising
- Template matching
- Image pyramids
- Bilateral filtering