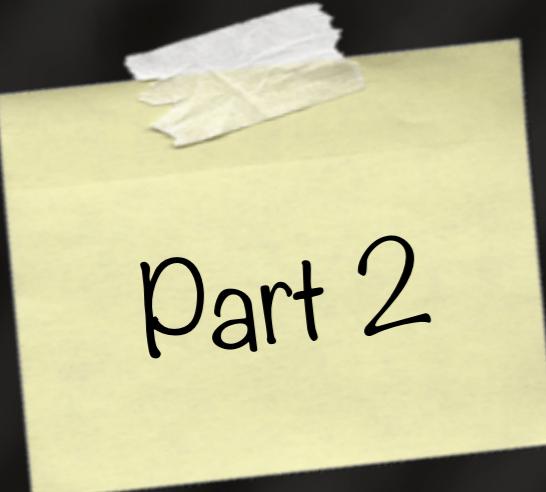


Intro to

Computer Vision

with Prof. Kosta Derpanis

Frequency analysis



Part 2

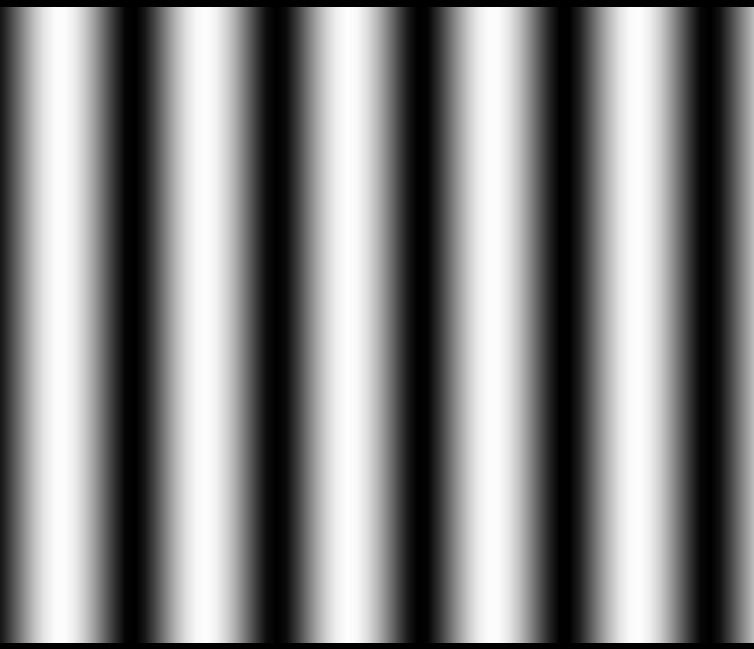
Ryerson
University

LECTURE TOPICS

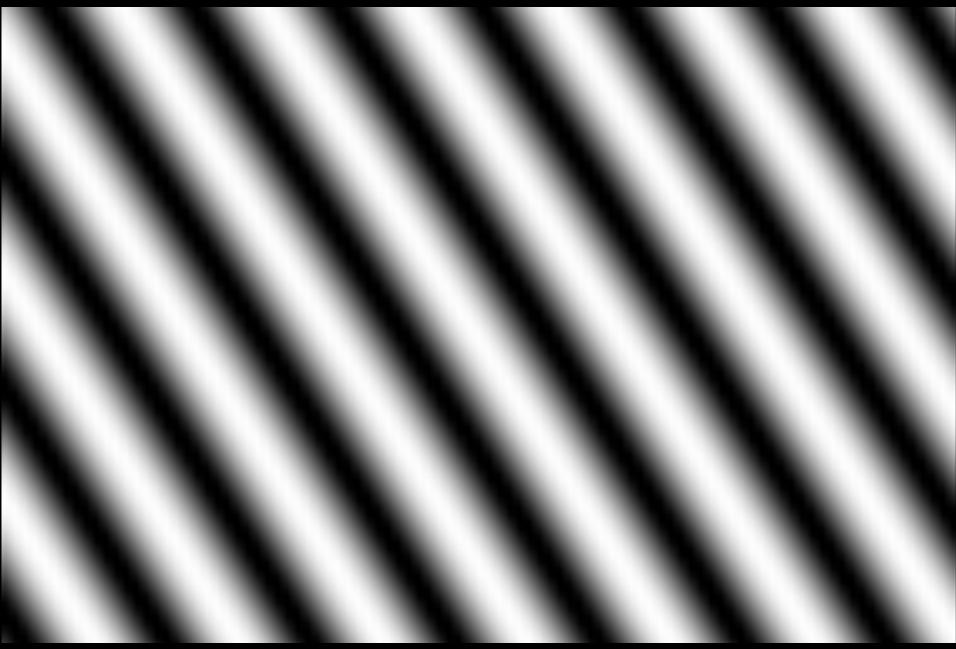
Fourier transform

Fourier properties

Signal aliasing



$+$ α_3



$+$ \dots

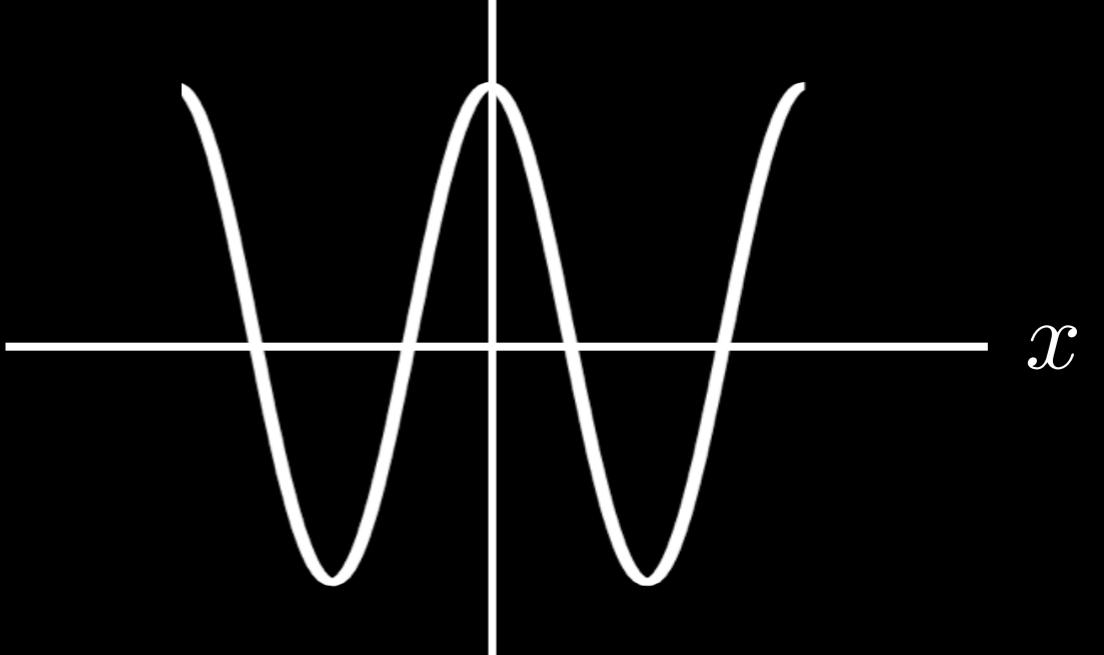
Inverse Discrete Fourier Transform

$$f[x, y] = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F[u, v] e^{2\pi i (x \frac{u}{M} + y \frac{v}{N})}$$

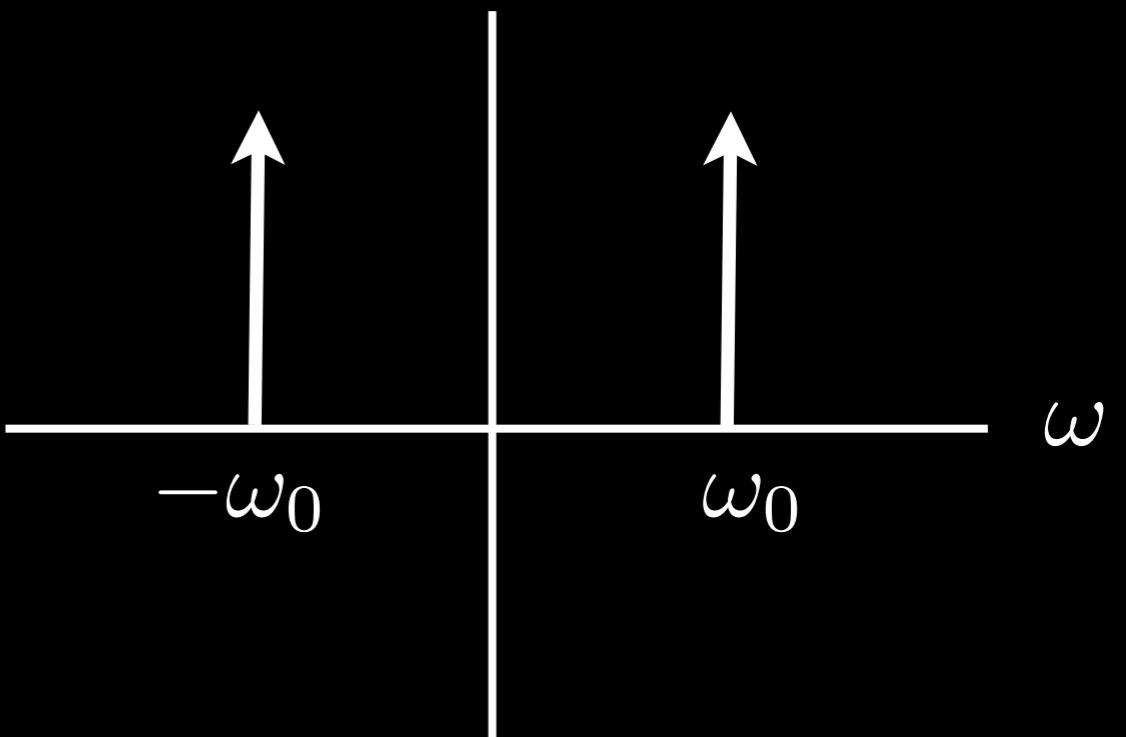
where

$$\begin{aligned} u &= 0, \dots, M - 1 \\ v &= 0, \dots, N - 1 \end{aligned}$$

$$\cos(2\pi\omega_0 x)$$

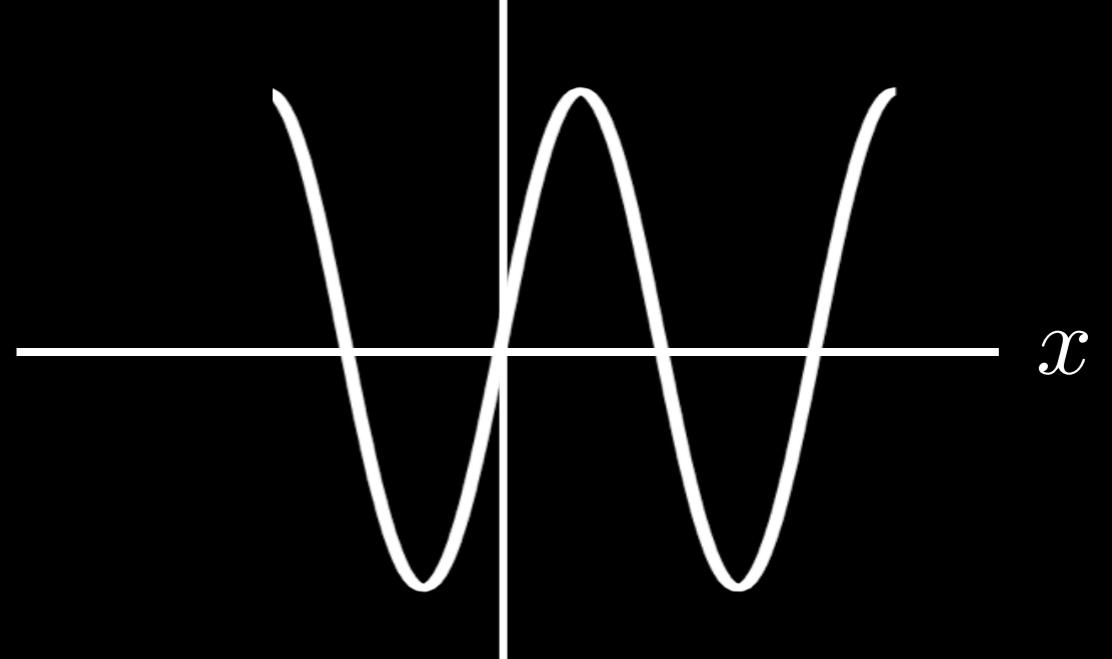


$$\frac{1}{2} (\delta(\omega - \omega_0) + \delta(\omega + \omega_0))$$

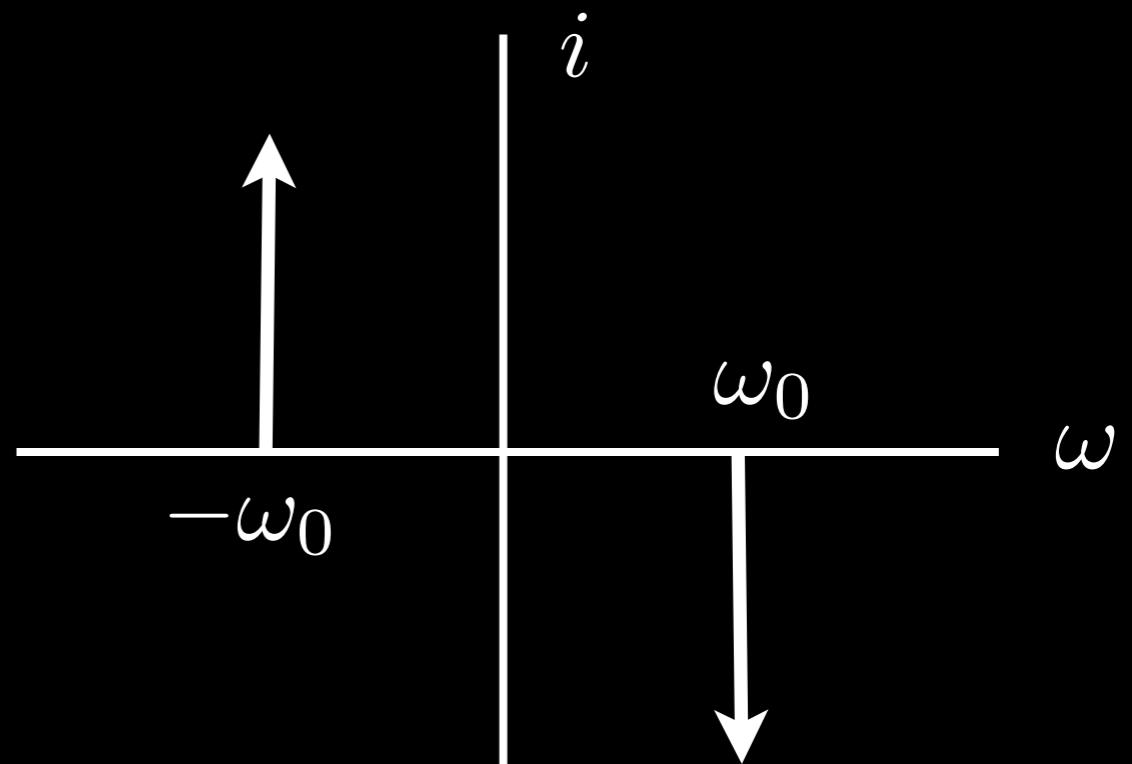


$$\cos(2\pi\omega_0 x) = \frac{1}{2} \left(e^{i2\pi\omega_0 x} + e^{-i2\pi\omega_0 x} \right)$$

$$\sin(2\pi\omega_0 x)$$

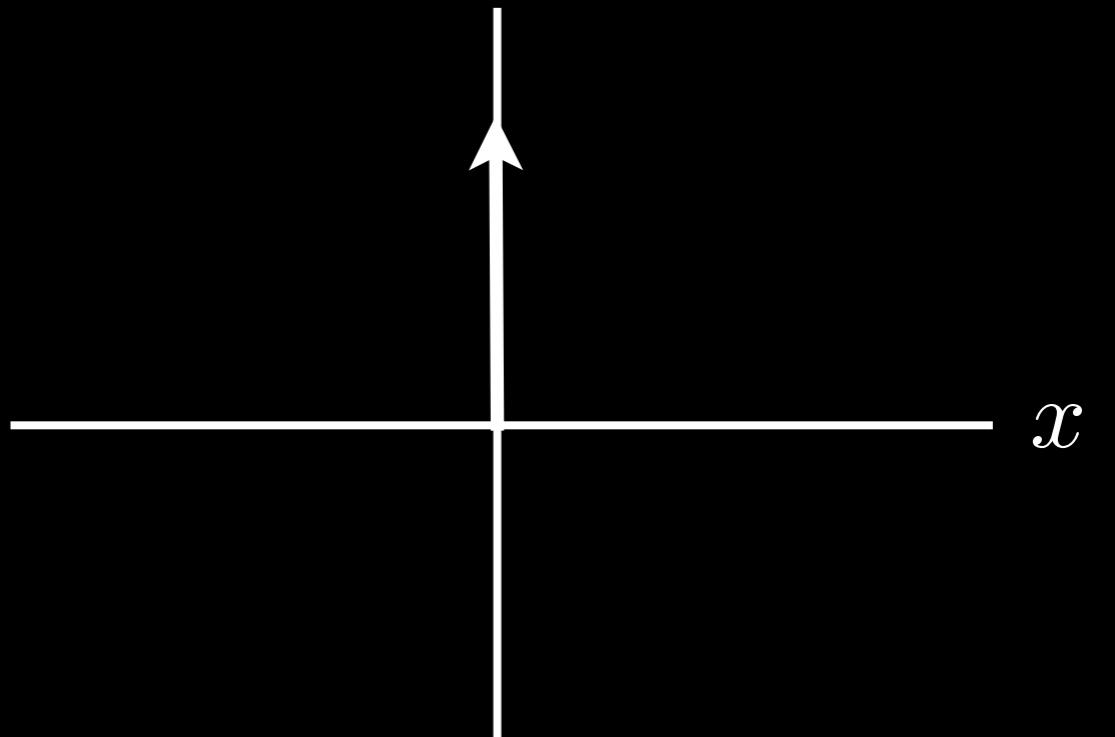


$$\frac{1}{2}i(\delta(\omega + \omega_0) - \delta(\omega - \omega_0))$$

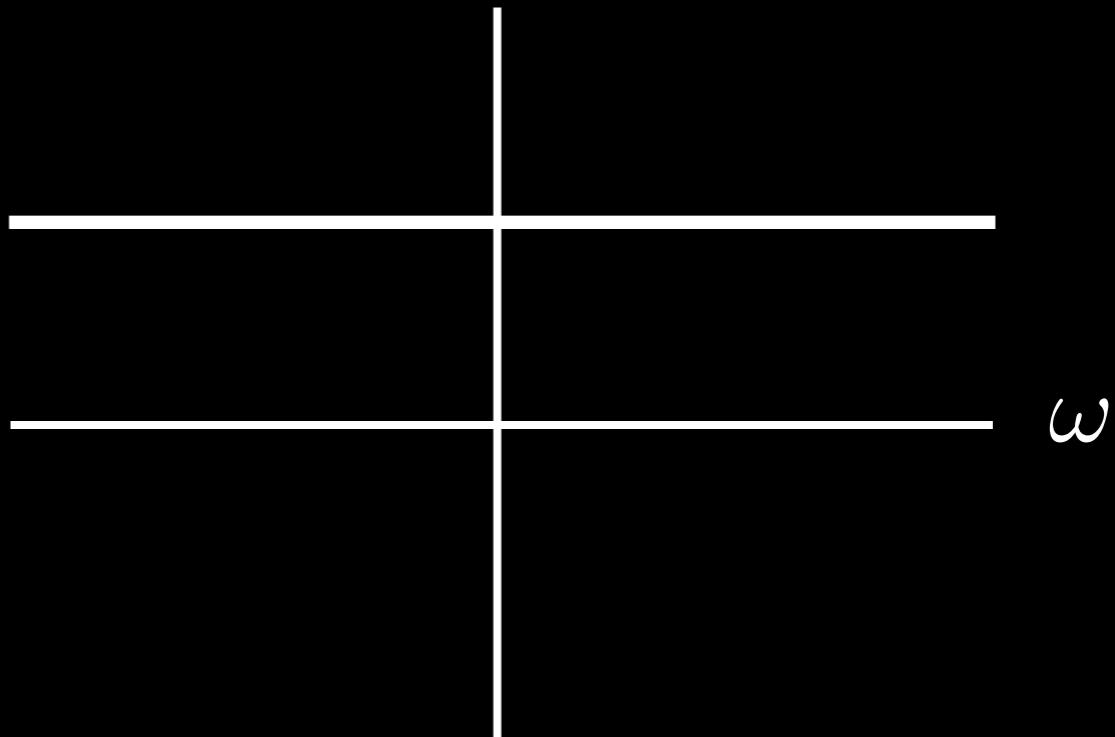


$$\sin(2\pi\omega_0 x) = \frac{1}{2i} \left(e^{i2\pi\omega_0 x} - e^{-i2\pi\omega_0 x} \right)$$

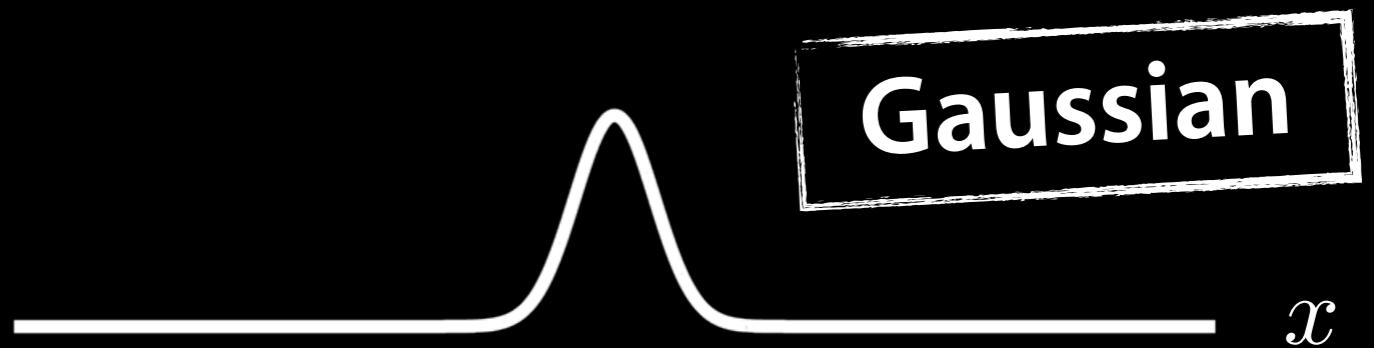
$$f(x) = \delta(x)$$



$$F(\omega) = 1$$

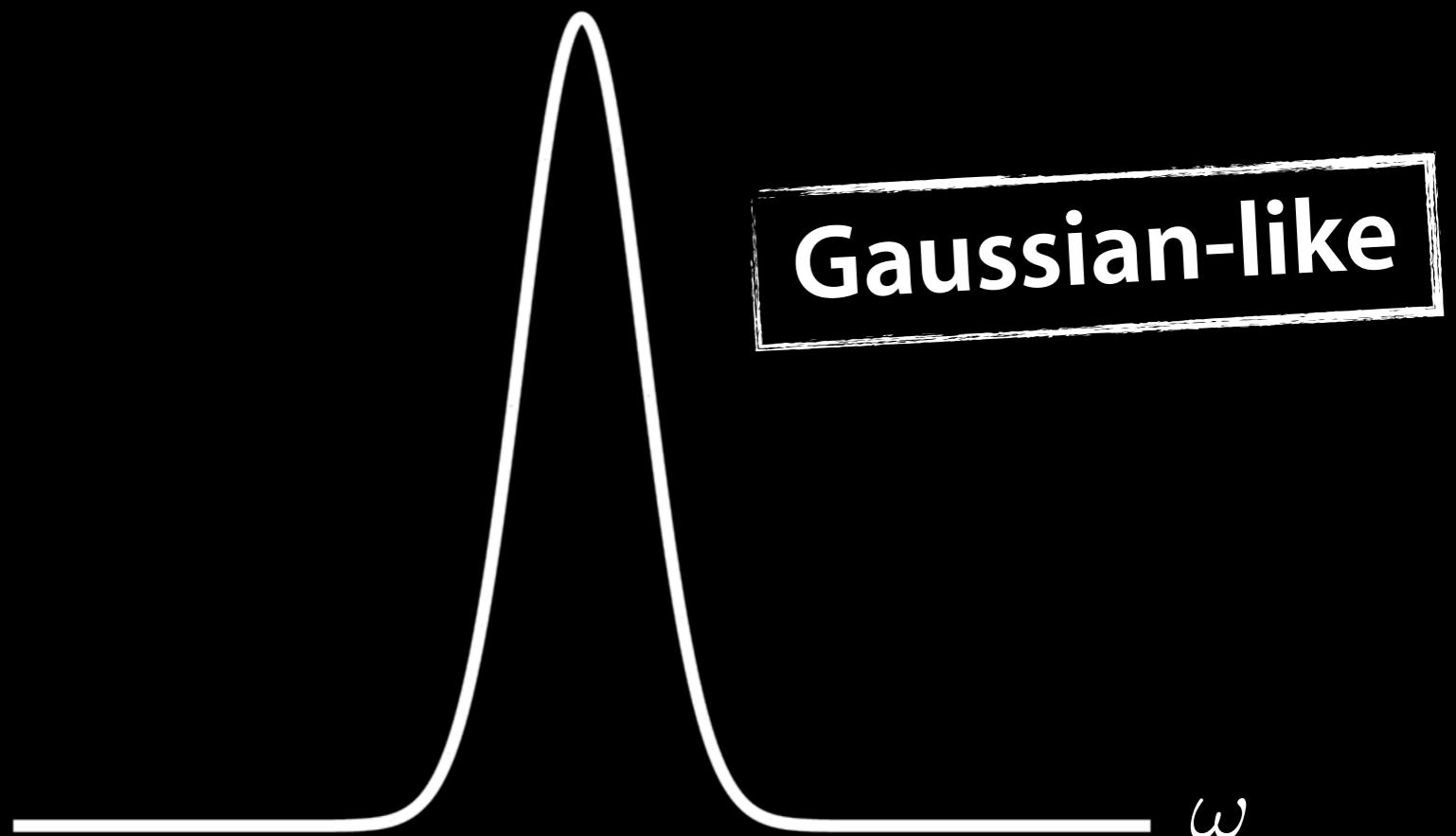


$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

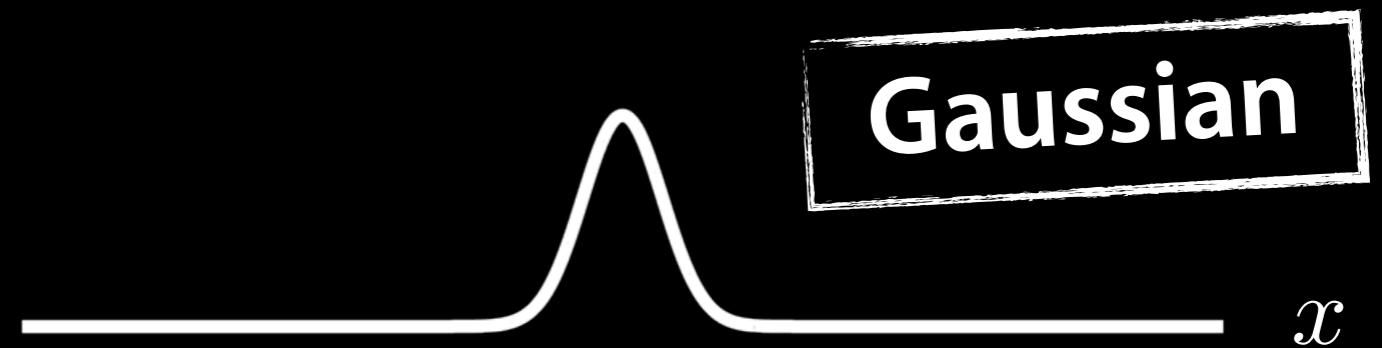


$$G(\omega) = e^{-\frac{(2\pi\omega)^2\sigma^2}{2}}$$

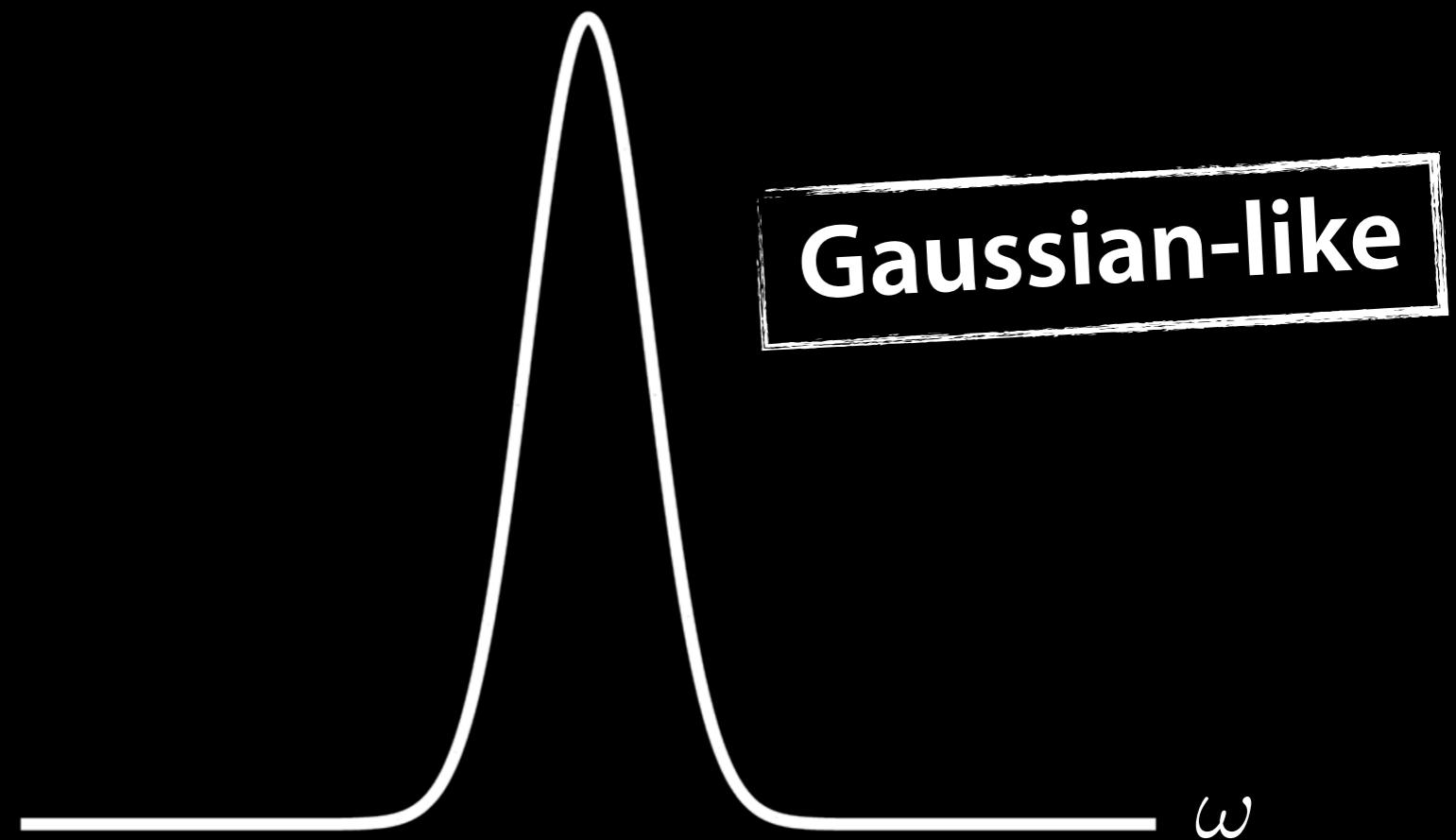
not normalized



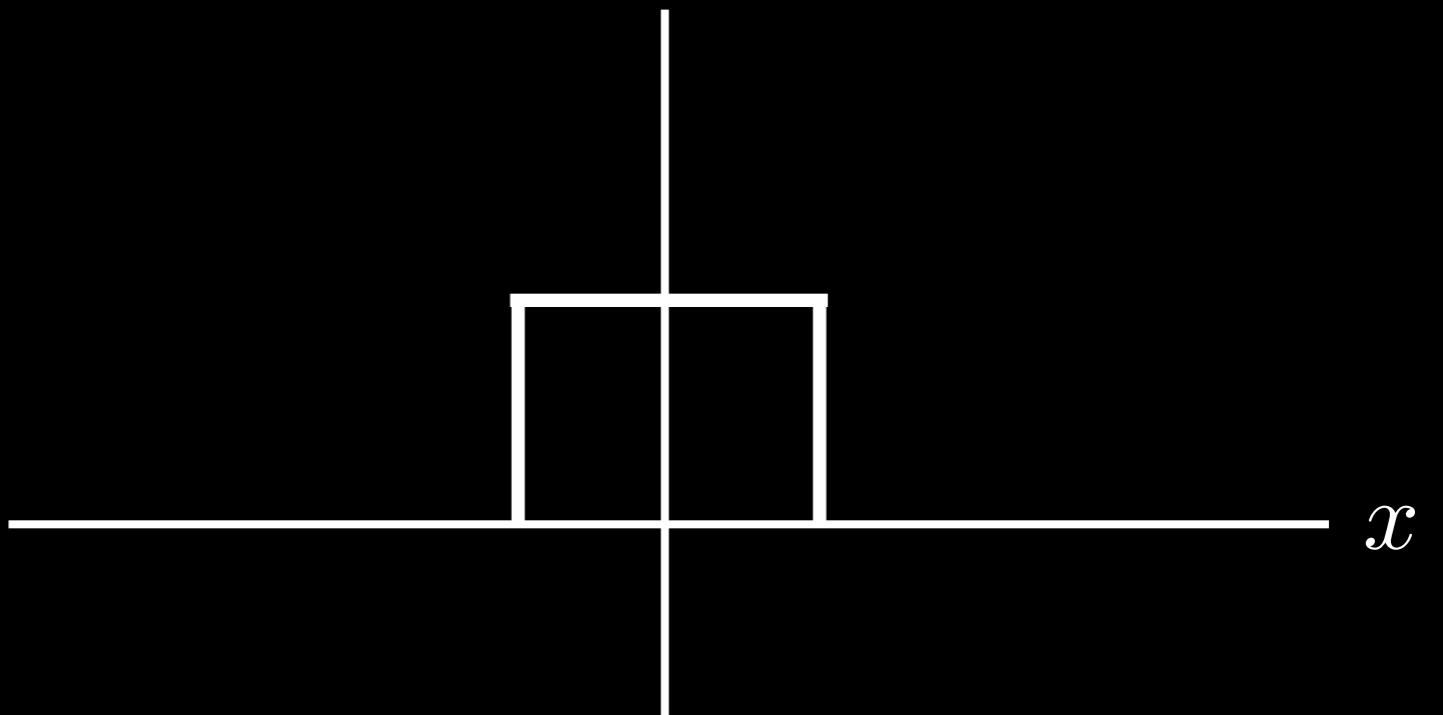
$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$



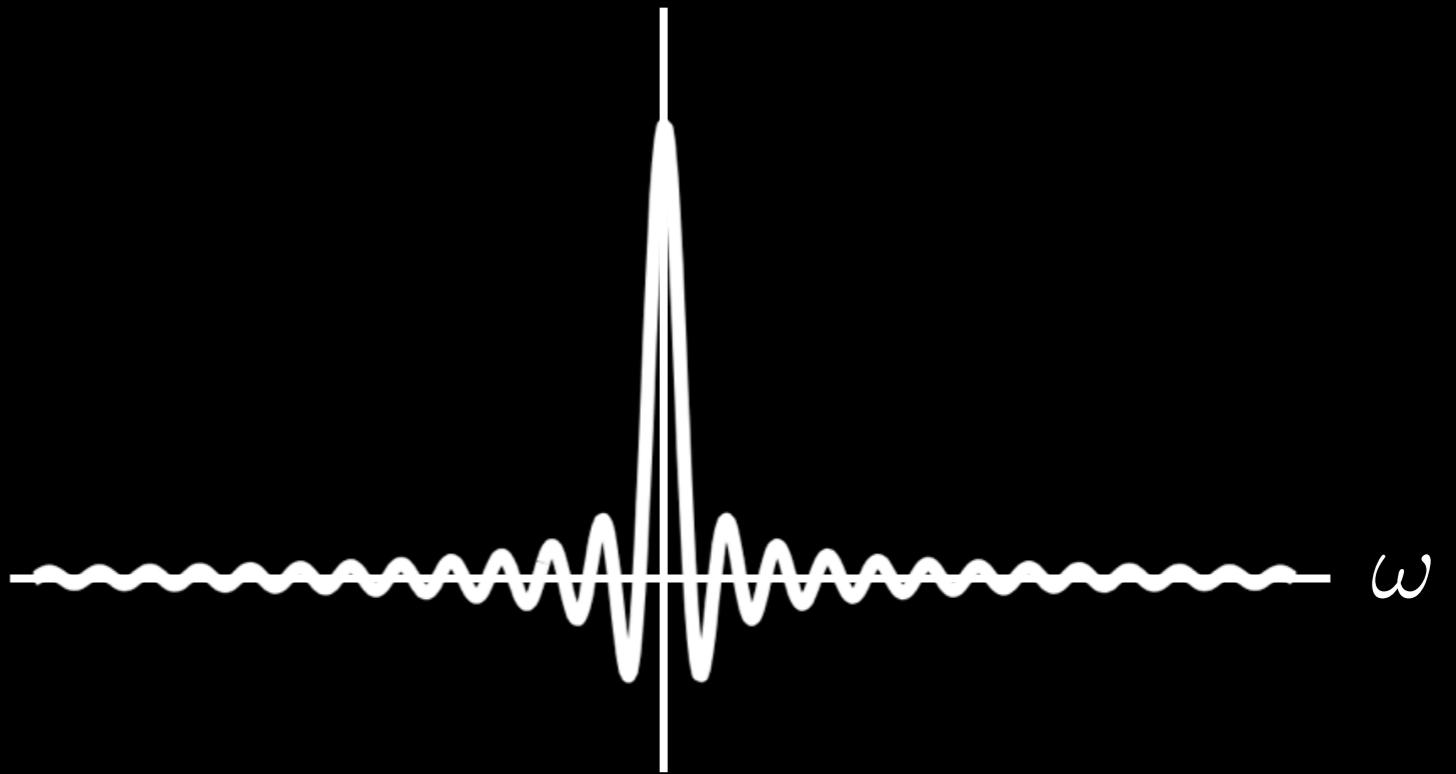
$$G(\omega) = e^{-\frac{(2\pi\omega)^2}{2}\sigma^2}$$



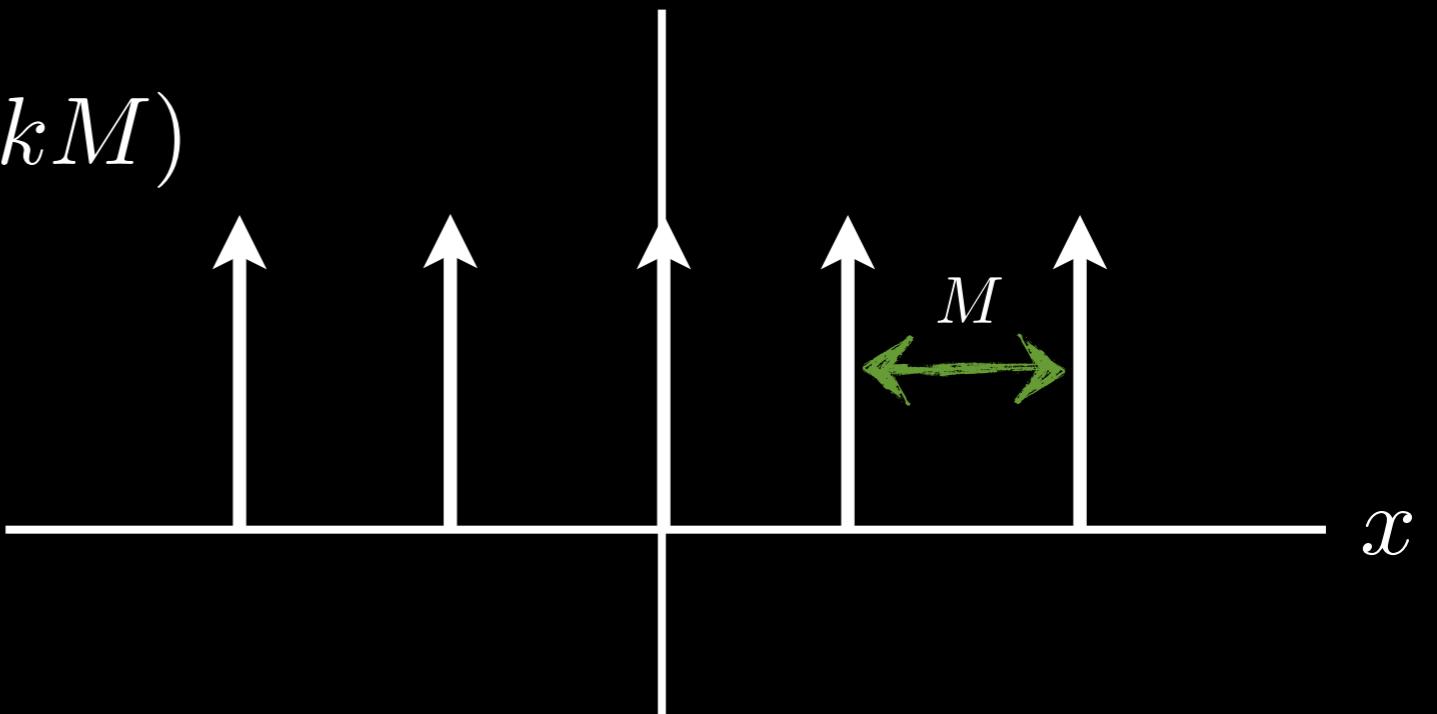
$\text{box}(x)$



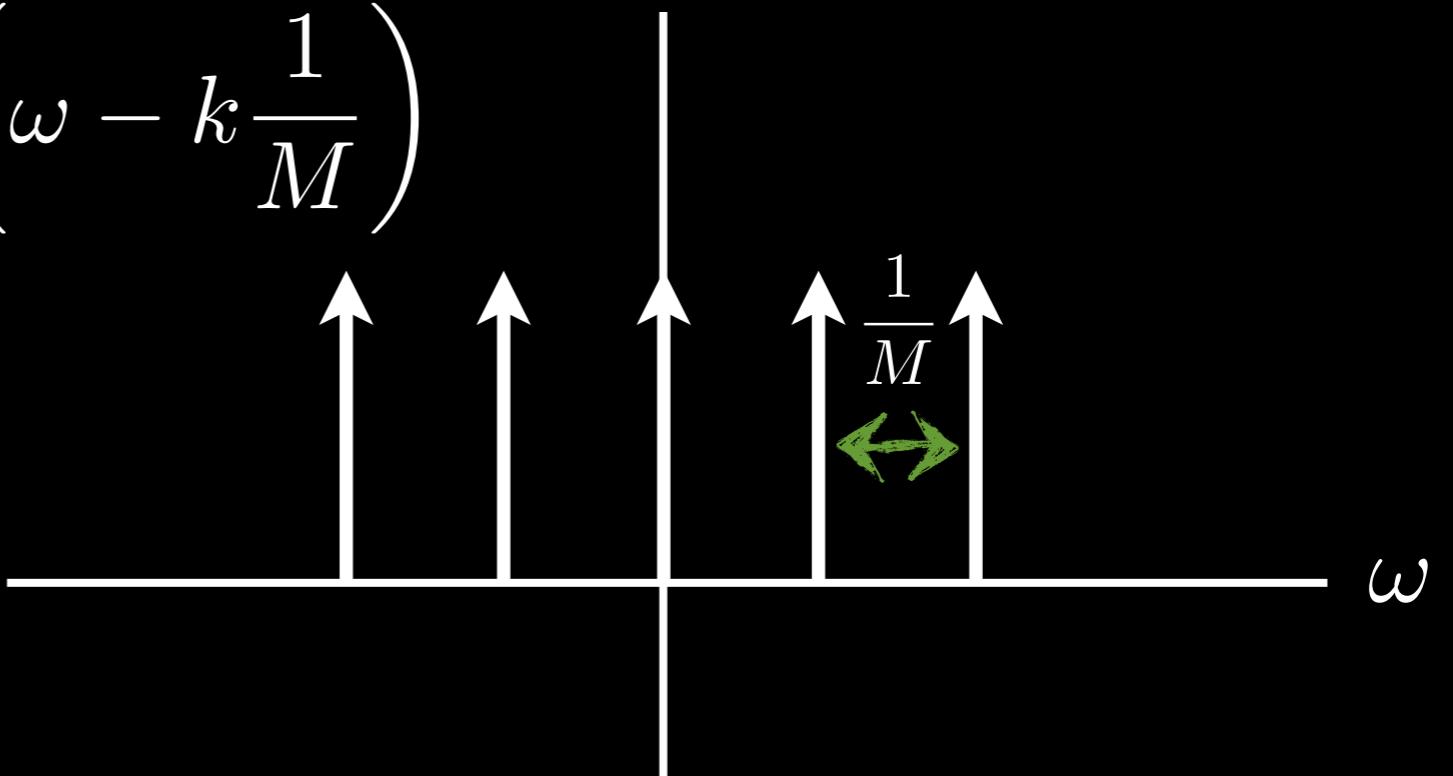
$\text{sinc}(\omega)$



$$\text{comb}(x) = \sum_{k=-\infty}^{\infty} \delta(x - kM)$$



$$\text{comb}(\omega) = \frac{1}{M} \sum_{k=-\infty}^{\infty} \delta\left(\omega - k \frac{1}{M}\right)$$



Fourier transform properties

Spatial domain

Frequency domain

Linearity	$c_1 f(x) + c_2 g(x)$	$c_1 F(\omega) + c_2 G(\omega)$
------------------	-----------------------	---------------------------------

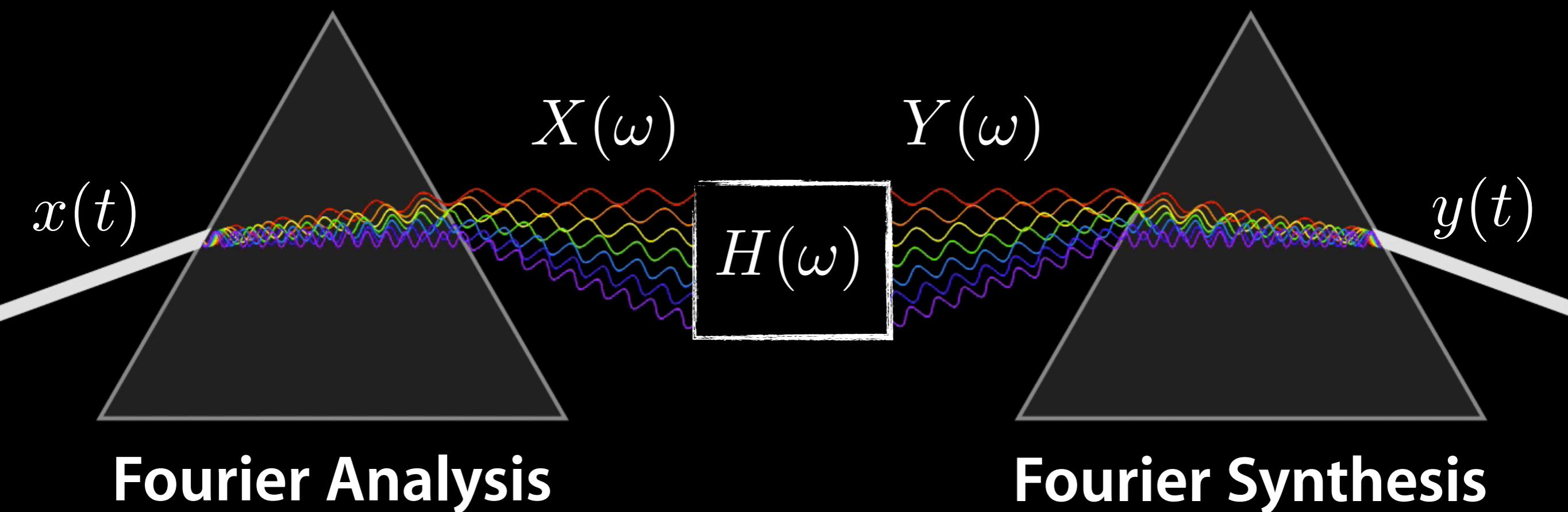
Scaling	$f(ax)$	$\frac{1}{ a } F\left(\frac{\omega}{a}\right)$
----------------	---------	--

Shifting	$f(x - x_0)$	$e^{-i2\pi\omega x_0} F(\omega)$
-----------------	--------------	----------------------------------

Differentiation	$\frac{d^n f(x)}{dx^n}$	$(i2\pi\omega)^n F(\omega)$
------------------------	-------------------------	-----------------------------

Convolution	$f(x) * g(x)$	$F(\omega)G(\omega)$
--------------------	---------------	----------------------

Convolution
Theorem



$$Y(\omega) = H(\omega)X(\omega)$$

$O(N^2)$

convolution

vs.

 $O(N \log N)$

FFT-based convolution

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

$$\mathcal{F}\{f * g\}(\omega)$$

$$= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} f(u)g(t-u)du \right] e^{-i2\pi\omega t} dt$$

convolution

Proof:
Convolution
Theorem

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

Proof:
Convolution
Theorem

$$\mathcal{F}\{f * g\}(\omega)$$

$$= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} f(u)g(t-u)du \right] e^{-i2\pi\omega t} dt$$

Fourier transform

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

Proof:
Convolution
Theorem

$$\mathcal{F}\{f * g\}(\omega)$$

$$= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} f(u)g(t-u)du \right] e^{-i2\pi\omega t} dt$$

change order of integration

$$= \int_{-\infty}^{\infty} f(u) \left[\int_{-\infty}^{\infty} g(t-u)e^{-i2\pi\omega t} dt \right] du$$

Does this look familiar?

Fourier
Transform
Properties

Spatial domain

Shifting

$$f(x - x_0)$$

Frequency domain

$$e^{-i2\pi\omega x_0} F(\omega)$$

Prove: $\mathcal{F}\{f(x) * g(x)\}(\omega) = F(\omega)G(\omega)$

Proof:
Convolution
Theorem

$$\mathcal{F}\{f * g\}(\omega)$$

$$= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} f(u)g(t-u)du \right] e^{-i2\pi\omega t} dt$$

change order of integration

$$= \int_{-\infty}^{\infty} f(u) \left[\int_{-\infty}^{\infty} g(t-u)e^{-i2\pi\omega t} dt \right] du$$

Does this look familiar?

$$e^{-i2\pi\omega u} G(\omega)$$

Prove: $\mathcal{F}\{f(x) * g(x)\}(\omega) = F(\omega)G(\omega)$

Proof:
Convolution
Theorem

$$\mathcal{F}\{f * g\}(\omega)$$

$$= \int_{-\infty}^{\infty} f(u) \left[\int_{-\infty}^{\infty} g(t-u) e^{-i2\pi\omega t} dt \right] du$$

shift theorem

$$= \int_{-\infty}^{\infty} f(u) e^{-i2\pi\omega u} G(\omega) du$$

$$e^{-i2\pi\omega u} G(\omega)$$

factor out

$$= G(\omega) \int_{-\infty}^{\infty} f(u) e^{-i2\pi\omega u} du$$

Prove: $\mathcal{F}\{f(x) * g(x)\}(\omega) = F(\omega)G(\omega)$

Proof:
Convolution
Theorem

$$\mathcal{F}\{f * g\}(\omega)$$

$$= \int_{-\infty}^{\infty} f(u) \left[\int_{-\infty}^{\infty} g(t-u)e^{-i2\pi\omega t} dt \right] du$$

$$= G(\omega) \int_{-\infty}^{\infty} f(u)e^{-i2\pi\omega u} du$$

Does this look familiar?

Fourier transform

Prove: $\mathcal{F}\{f(x) * g(x)\}(\omega) = F(\omega)G(\omega)$

Proof:
Convolution
Theorem

$$\mathcal{F}\{f * g\}(\omega)$$

$$= \int_{-\infty}^{\infty} f(u) \left[\int_{-\infty}^{\infty} g(t-u) e^{-i2\pi\omega t} dt \right] du$$

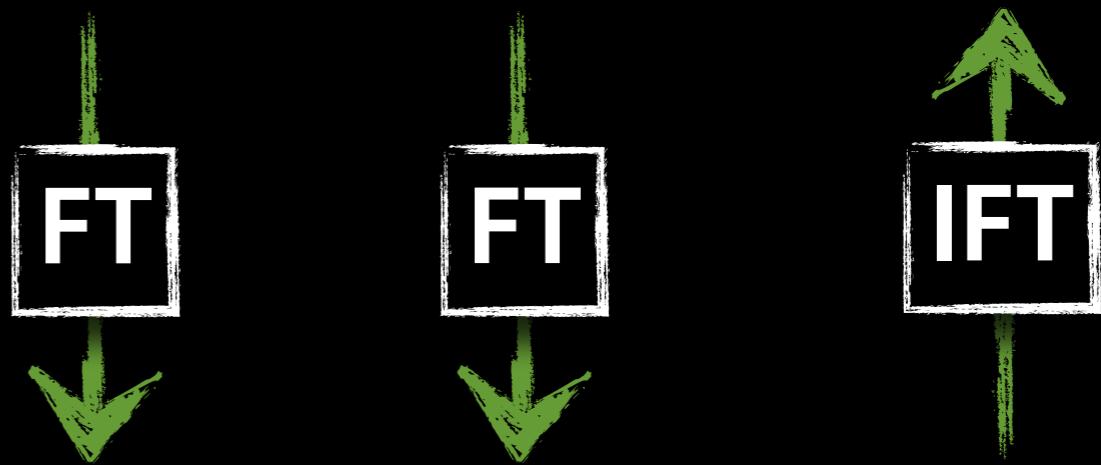
$$= G(\omega) \int_{-\infty}^{\infty} f(u) e^{-i2\pi\omega u} du$$

$$= F(\omega)G(\omega)$$

Convolution Theorem

Spatial domain

$$f * g = h$$



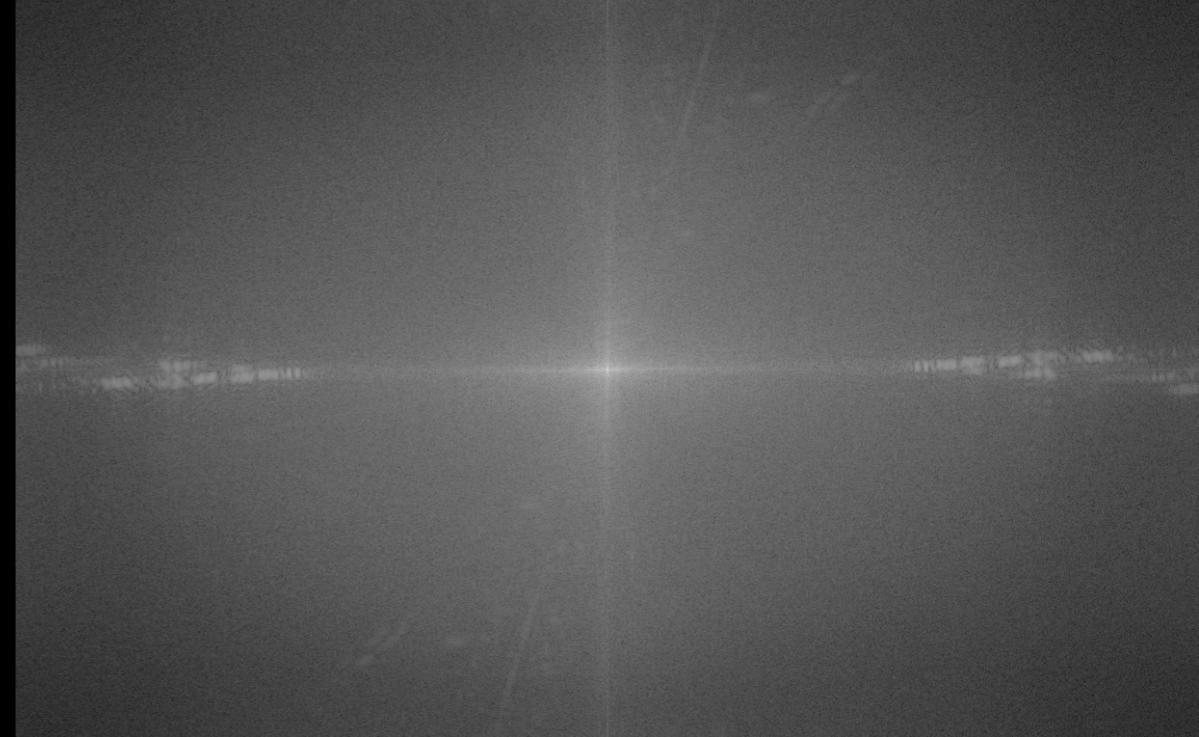
Frequency domain

$$F \times G = H$$

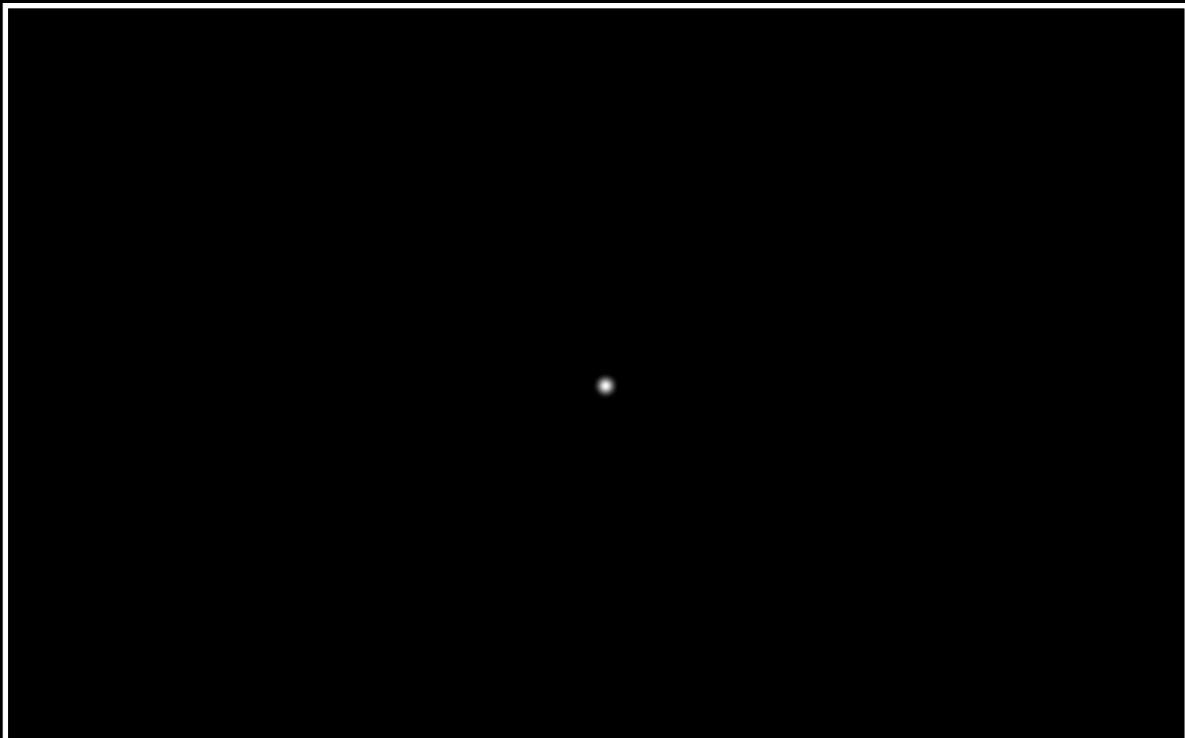
Spatial domain



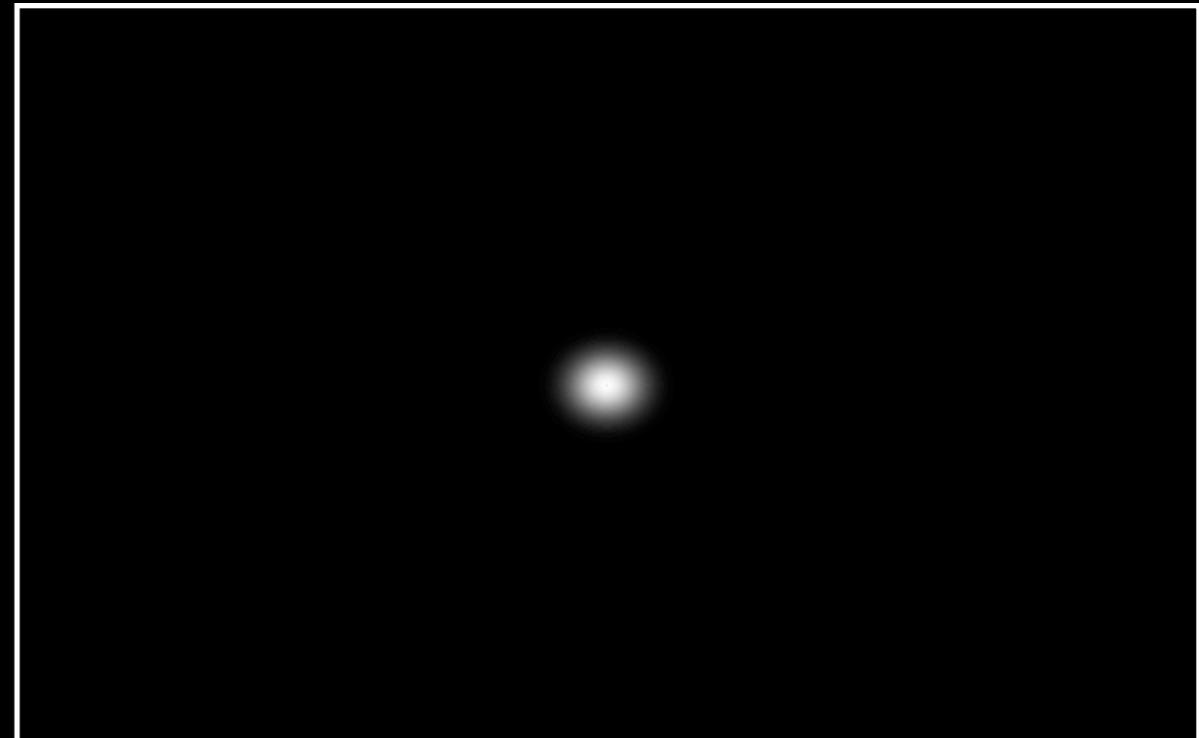
Frequency domain



*



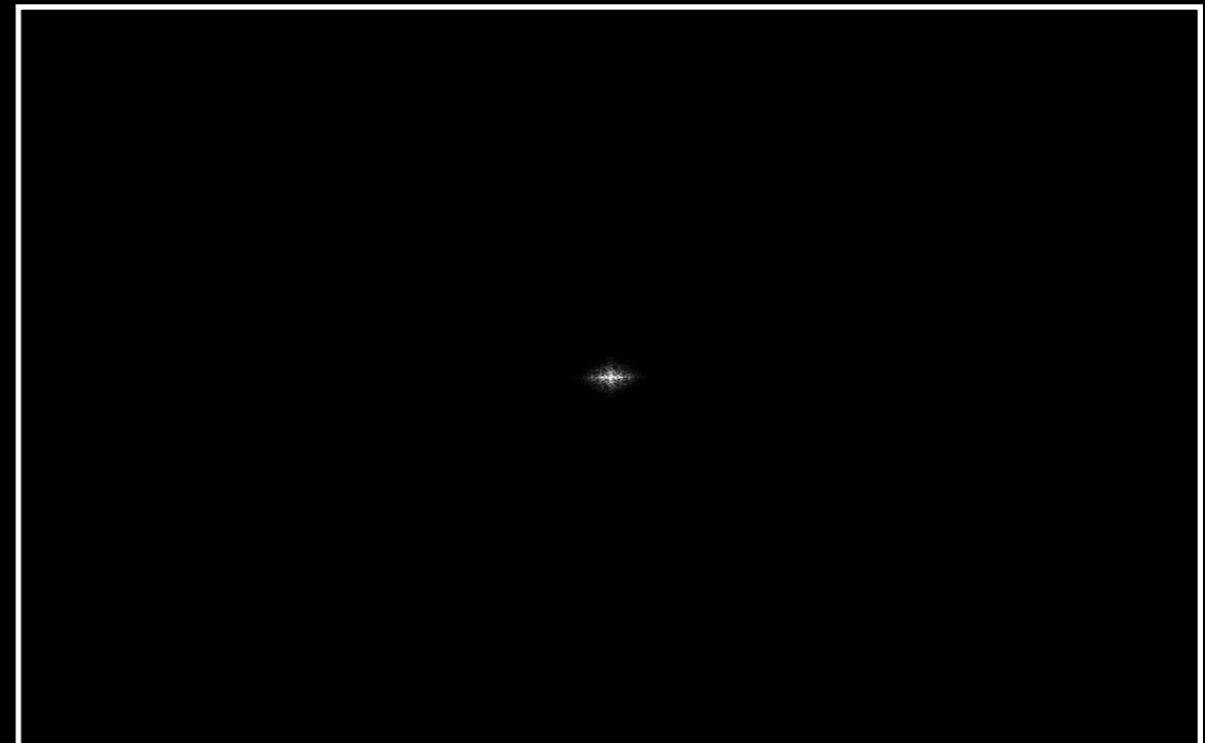
×



Spatial domain



Frequency domain



Frequency
Domain
Filtering

```
im = imread('toronto.jpg');
im = double(rgb2gray(im))/255;
[imh, imw] = size(im);

sigma = 10;
f = fspecial('gaussian', 2*sigma*3+1, sigma);
f = padarray(f, [imh imw]-(2*sigma*3+1), 'post');
f = circshift(f, -3*[sigma sigma]);

im_dft = fft2(im, size(im,1), size(im,2));
f_dft = fft2(f, size(im,1), size(im,2));

im_f_dft = im_dft .* f_dft;
im_f = ifft2(im_f_dft);

imshow(im_f, [])
```

```
im = imread('toronto.jpg');
im = double(rgb2gray(im))/255;
[imh, imw] = size(im);

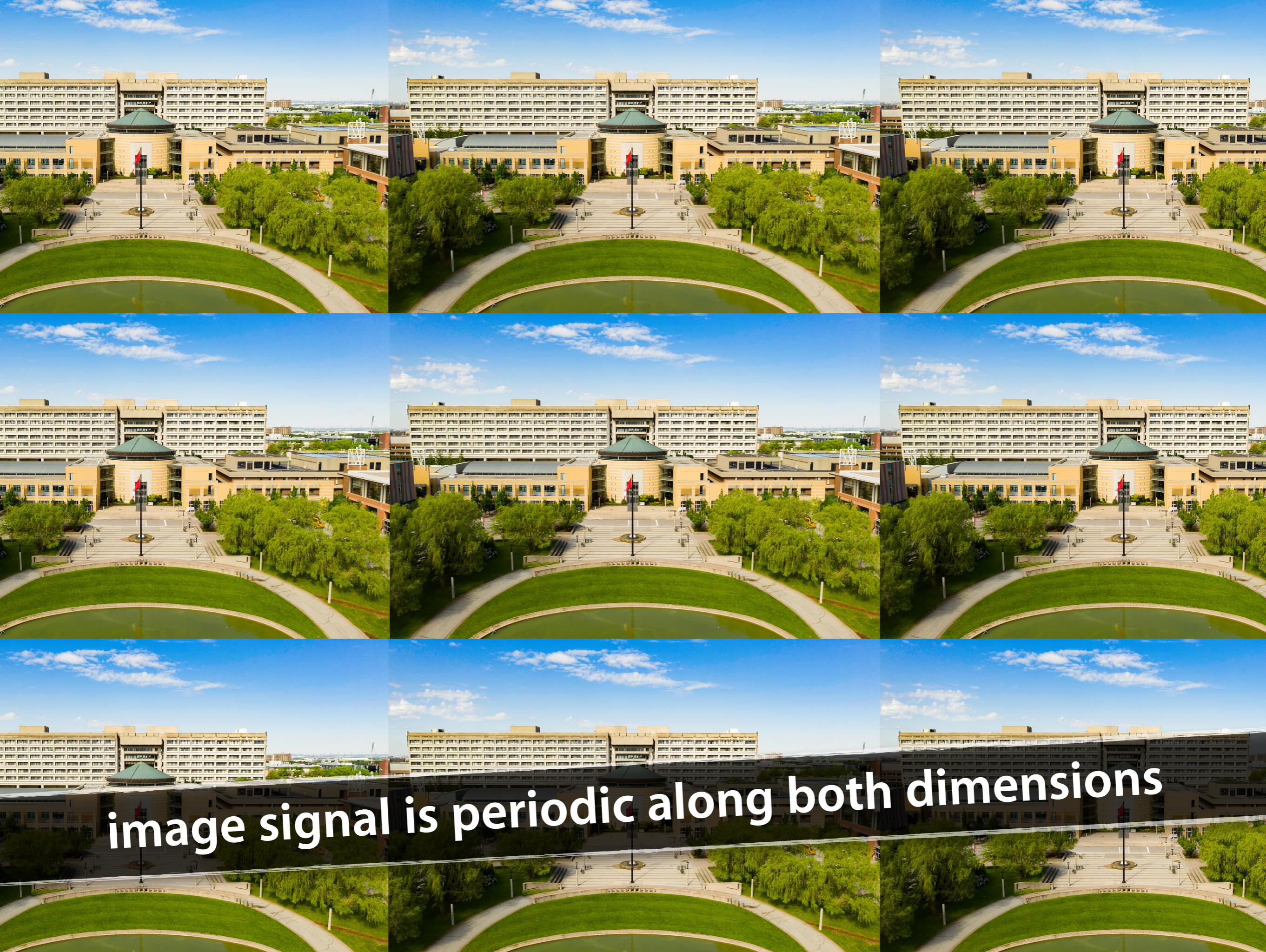
sigma = 10;
f = fspecial('gaussian', 2*sigma*3+1, sigma);
f = padarray(f, [imh imw]-(2*sigma*3+1), 'post');
f = circshift(f, -3*[sigma sigma]);
```

DFT assumption: image signal is periodic

```
_dft = fft2(I, size(im,1), size(im,2));
```

```
im_f_dft = im_dft .* f_dft;
im_f = ifft2(im_f_dft);
```

```
imshow(im_f, [])
```



```
im = imread('toronto.jpg');
im = double(rgb2gray(im))/255;
[imh, imw] = size(im);

sigma = 10;
f = fspecial('gaussian', 2*sigma*3+1, sigma);
f = padarray(f, [imh imw]-(2*sigma*3+1), 'post');
f = circshift(f, -3*[sigma sigma]);

im_dft = fft2(im, size(im,1), size(im,2));
f_dft = fft2(f, size(im,1), size(im,2));

im_f_dft = im_dft .* f_dft;
im_f = ifft2(im_f_dft, 'none');

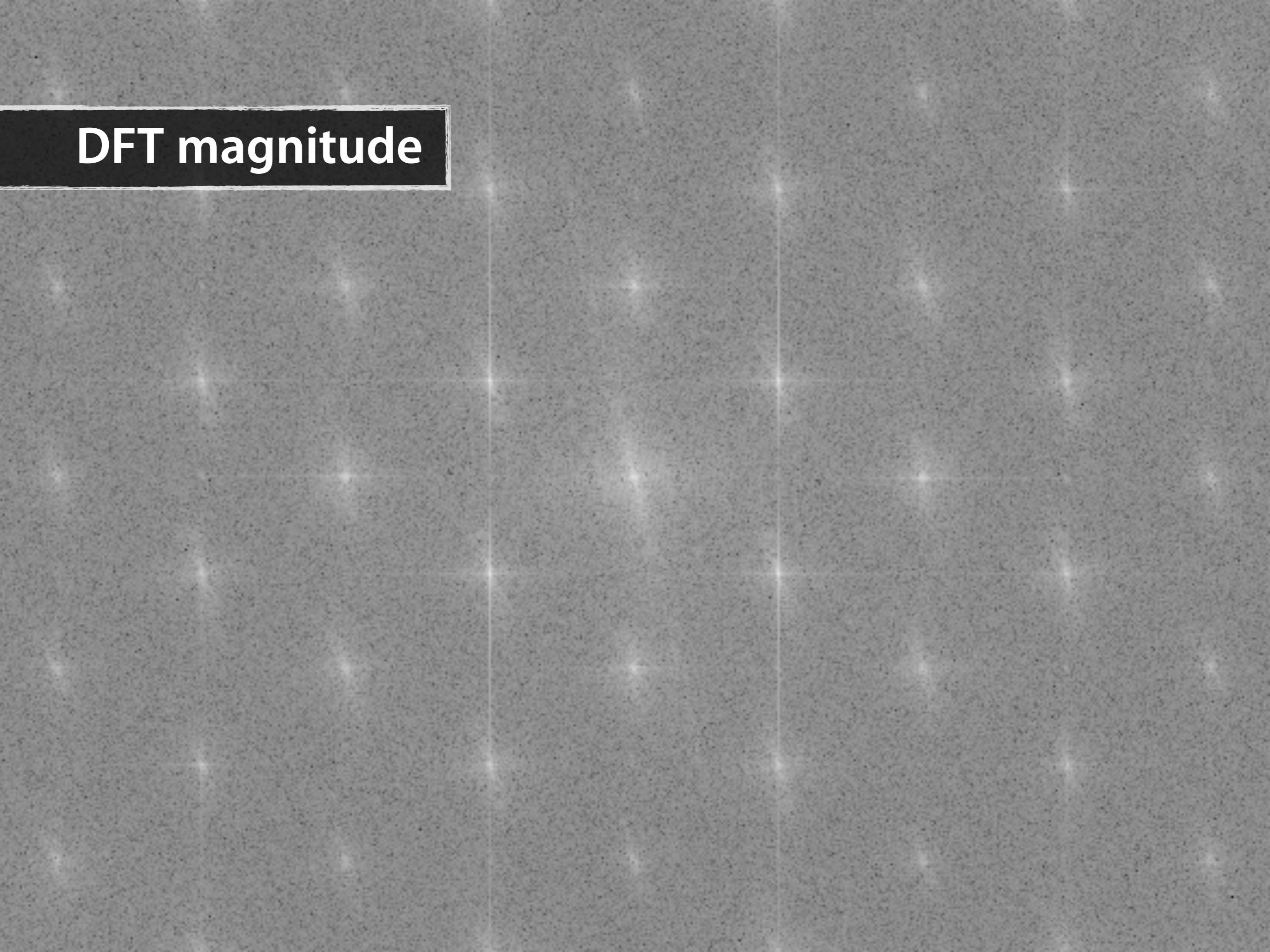
imshow(im_f, [])
```

pointwise multiplication

halftone image



DFT magnitude



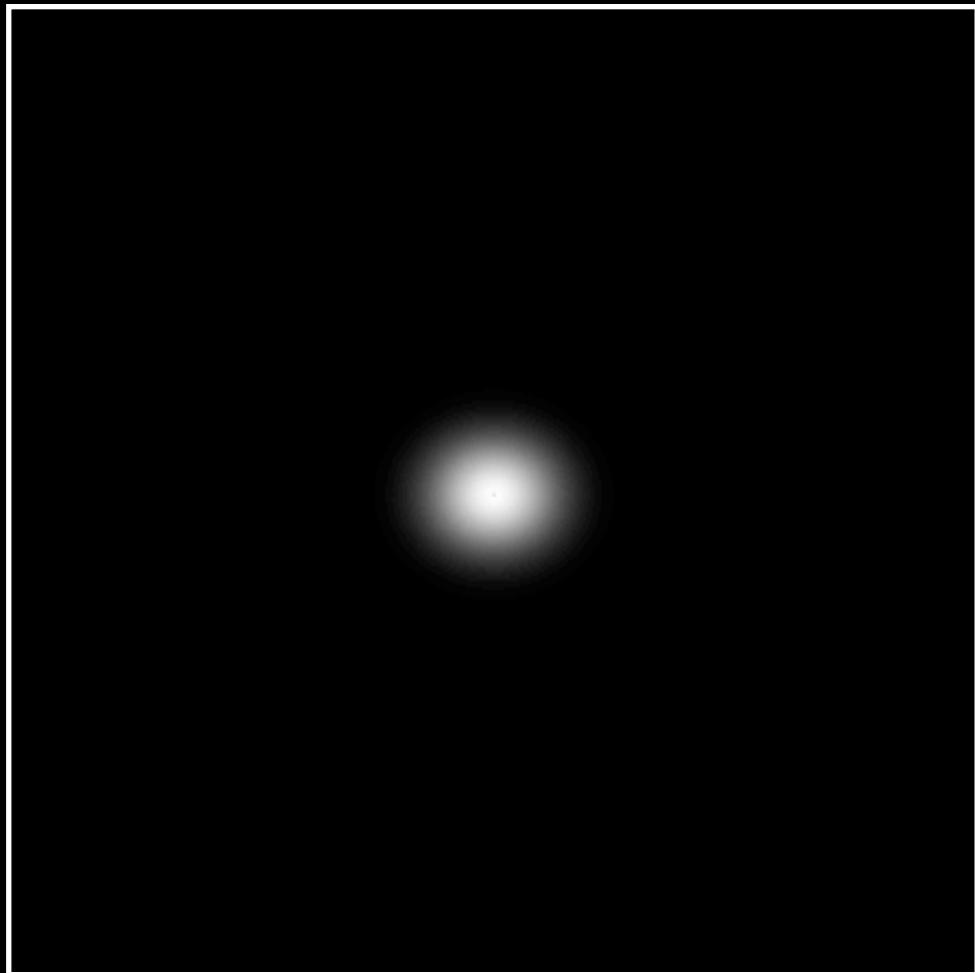
notch filtered spectrum

notch filtered image

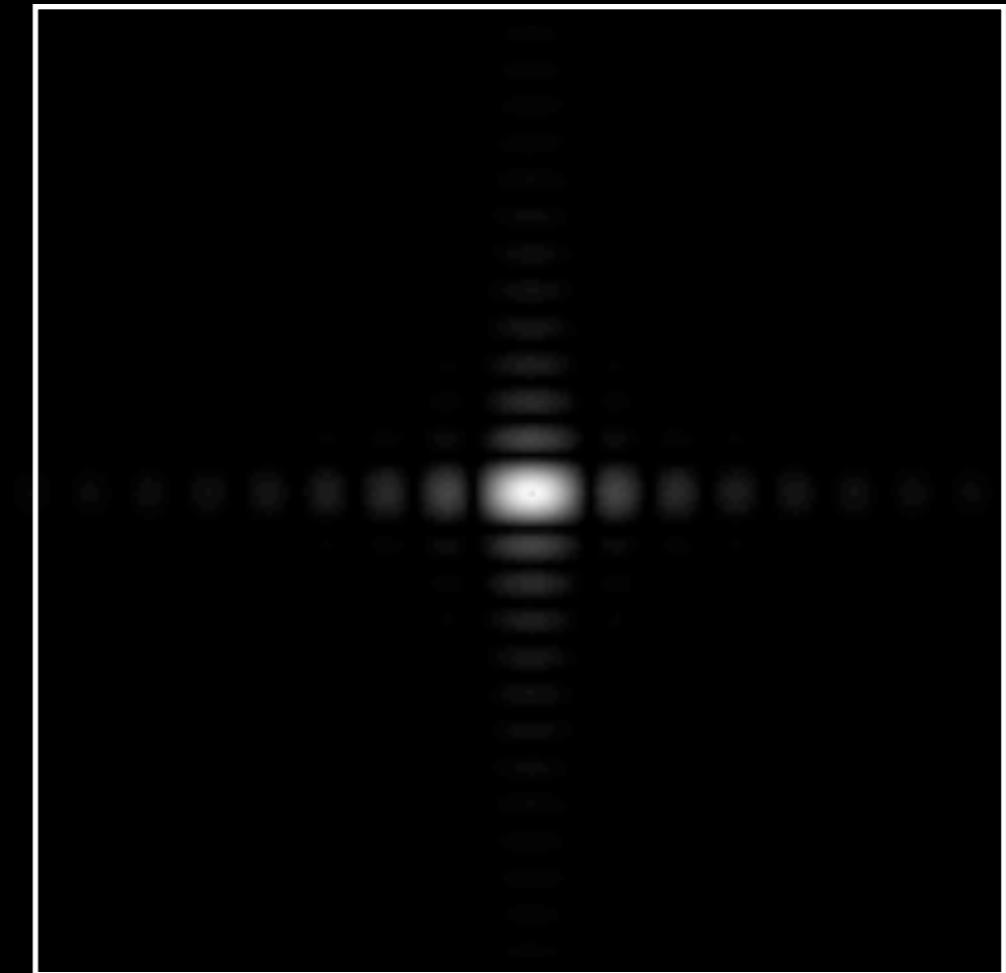




Why does the Gaussian filtered image appear smoother than the box filtered one?

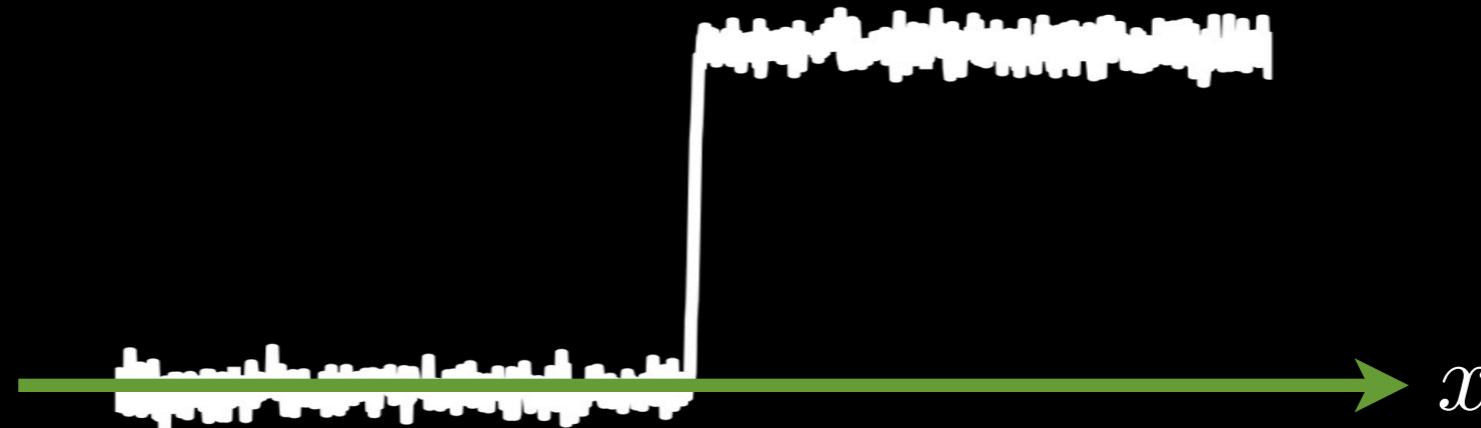


Gaussian Fourier magnitude



Box Fourier magnitude

$$f(x)$$

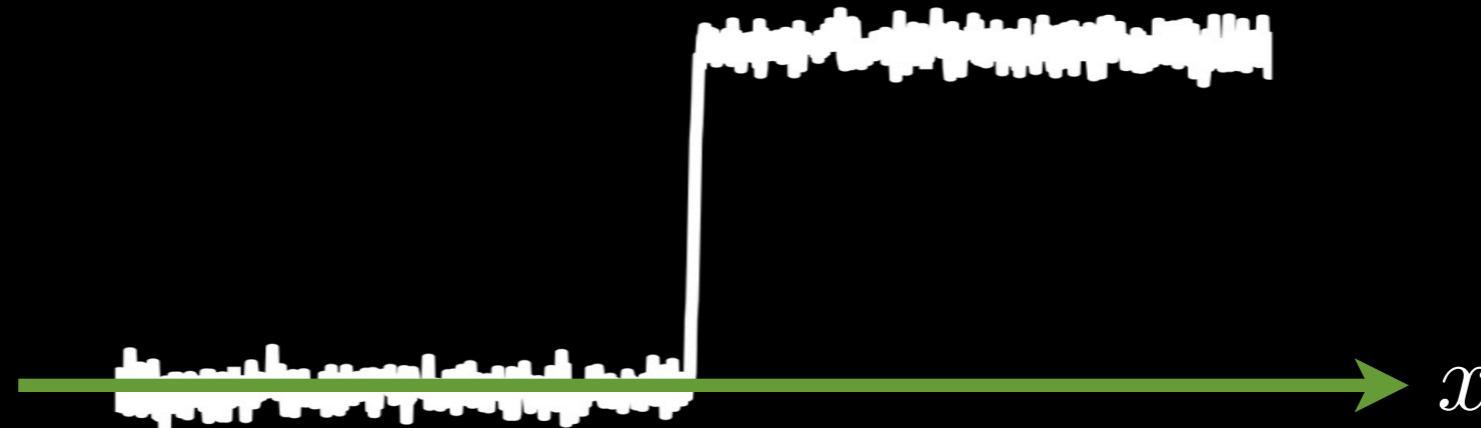


$$\frac{d}{dx} f(x)$$

Why does differentiation amplify noise?

$$\mathcal{F} \left\{ \frac{d^n f(x)}{dx^n} \right\} = (i2\pi\omega)^n F(\omega)$$

$$f(x)$$

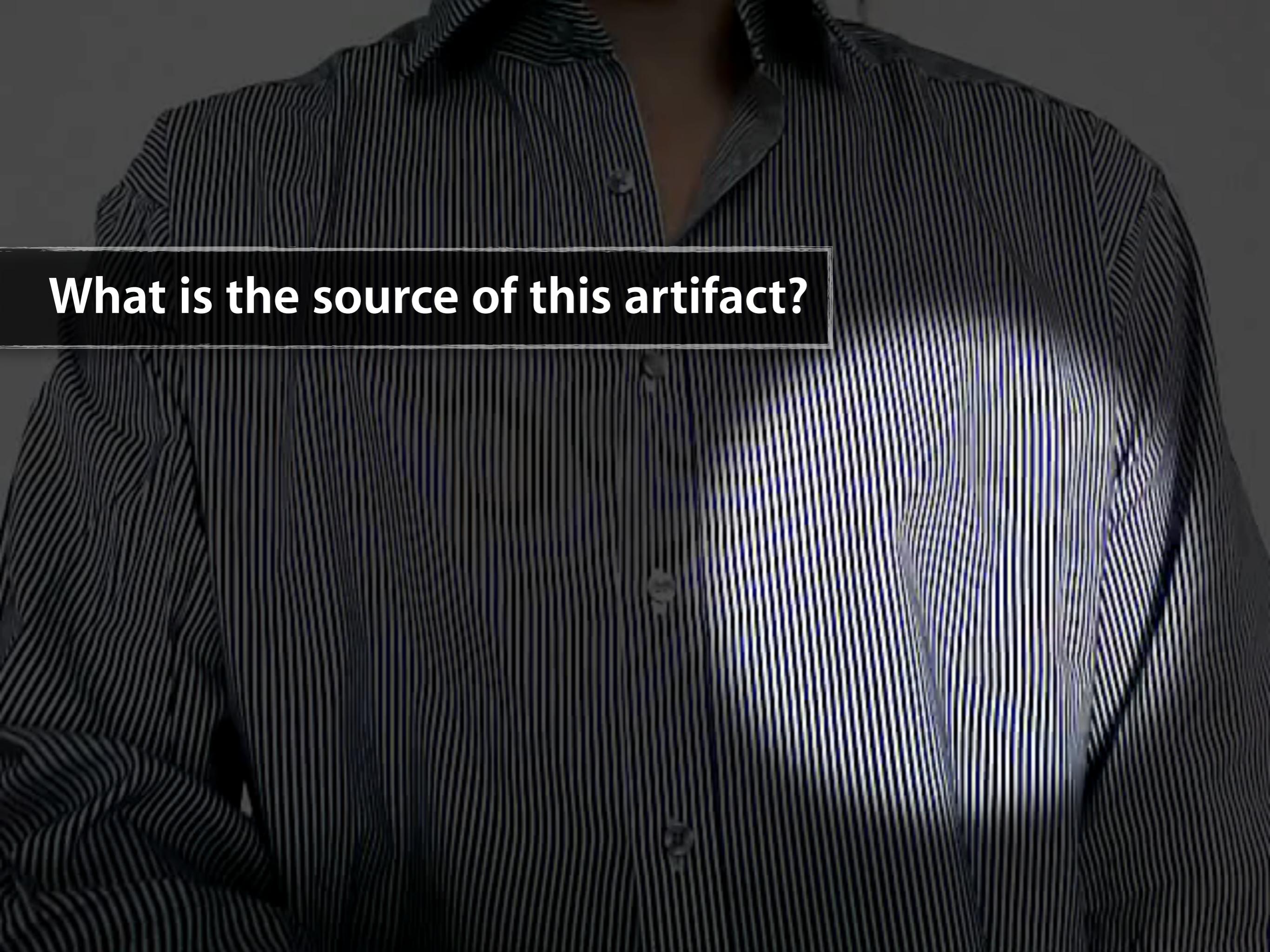


$$\frac{d}{dx} f(x)$$

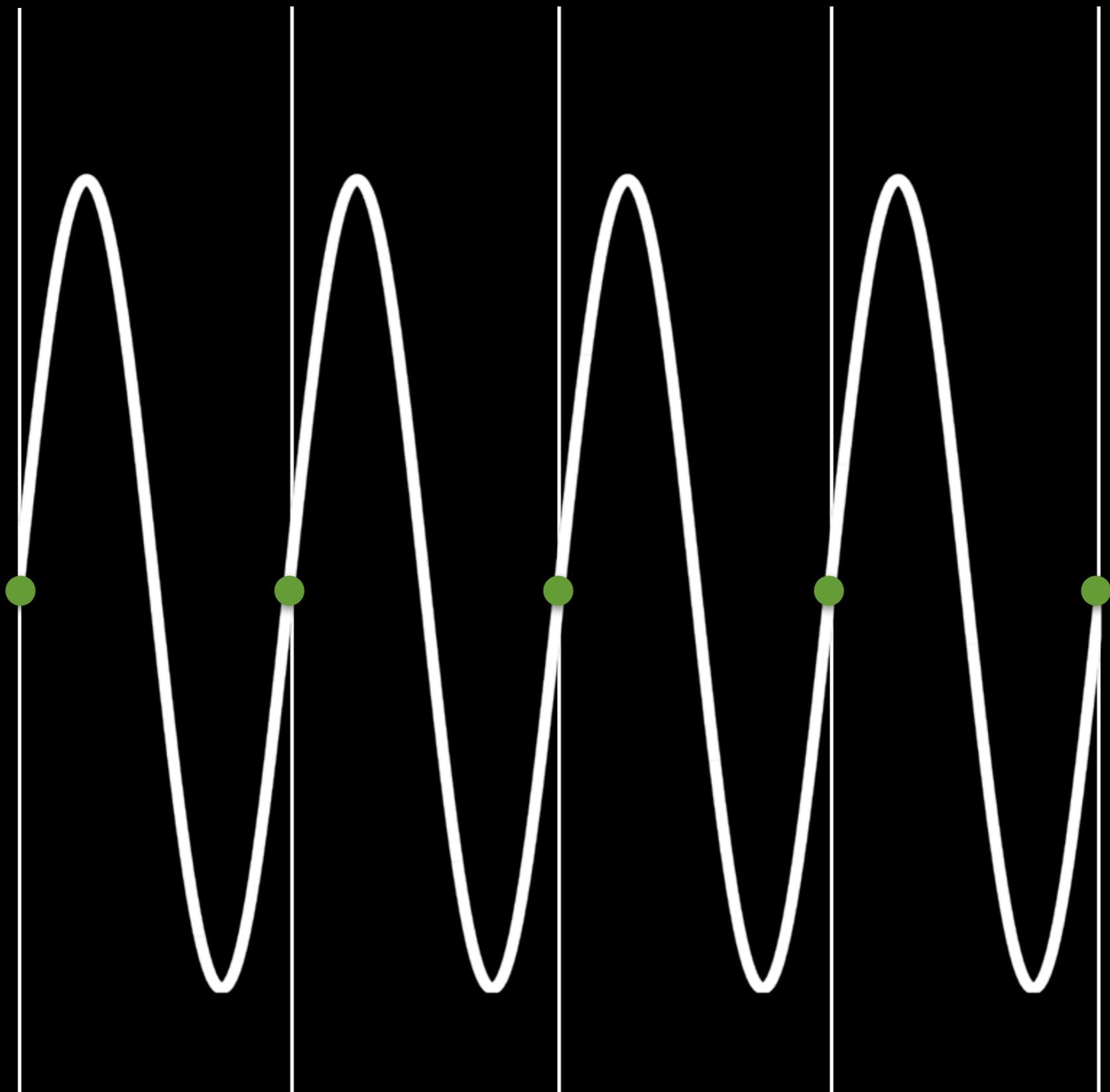


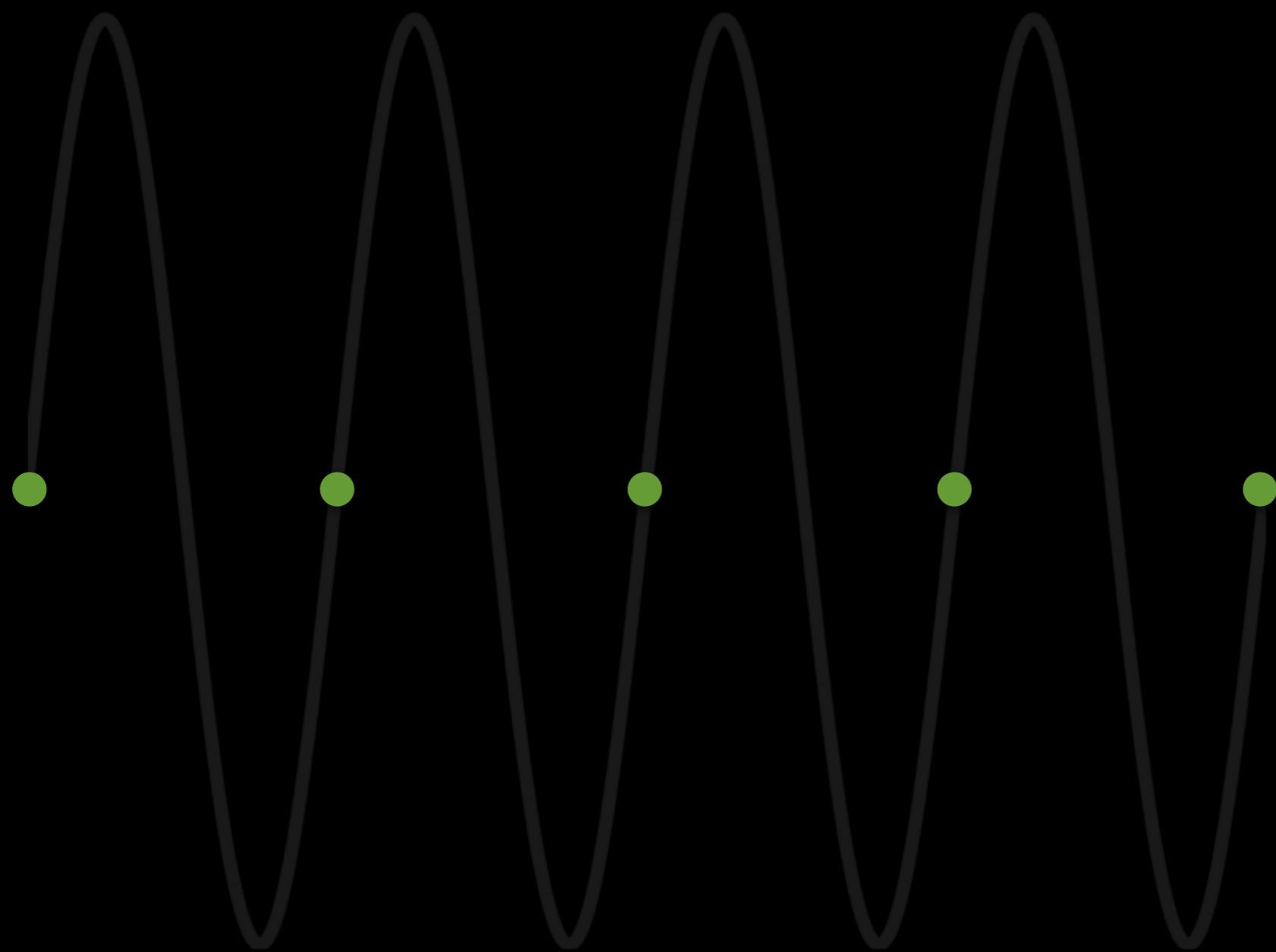
Why does differentiation amplify noise?

$$\mathcal{F} \left\{ \frac{d^n f(x)}{dx^n} \right\} = (i2\pi\omega)^n F(\omega)$$

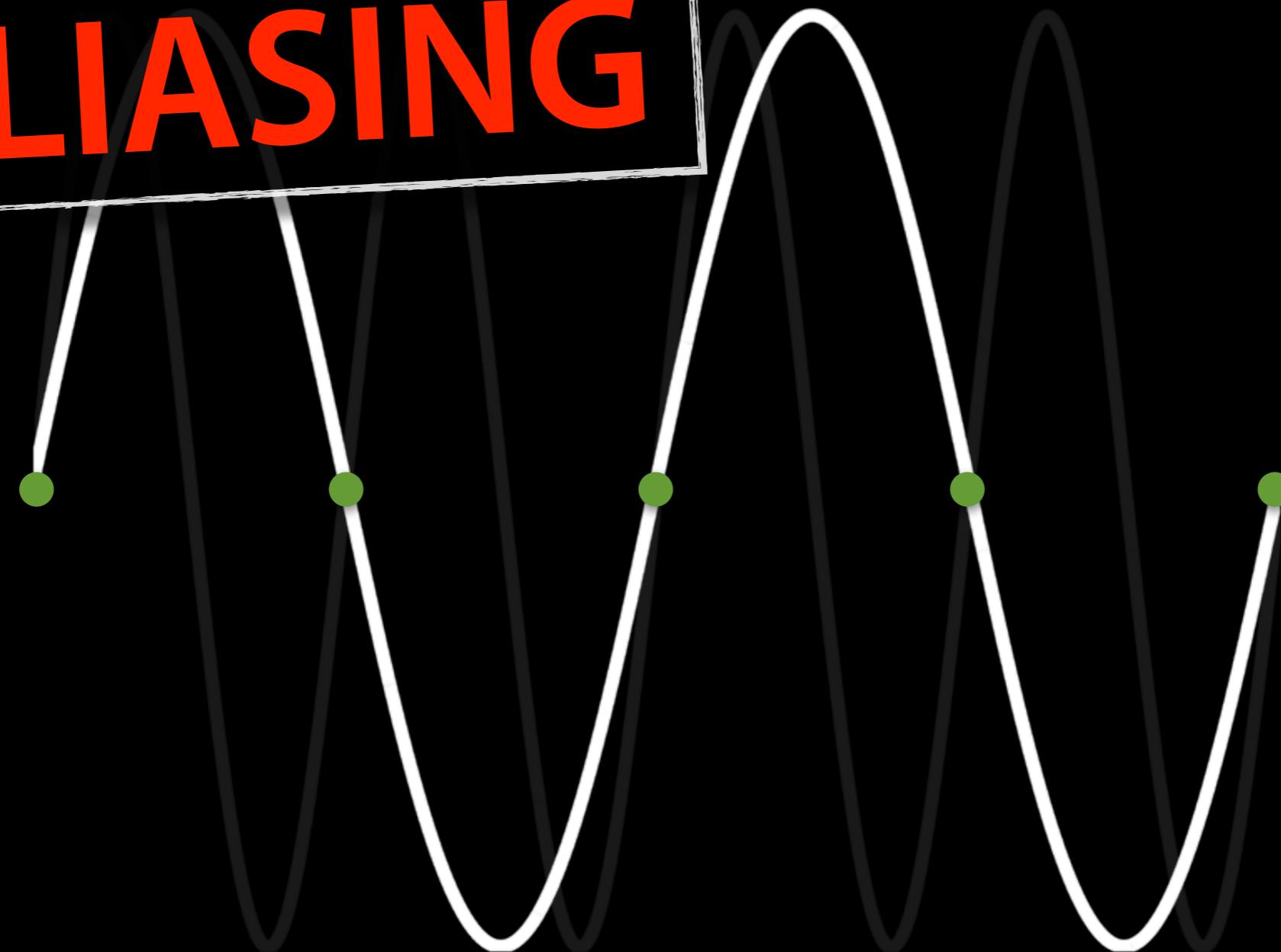


What is the source of this artifact?





ALIASING



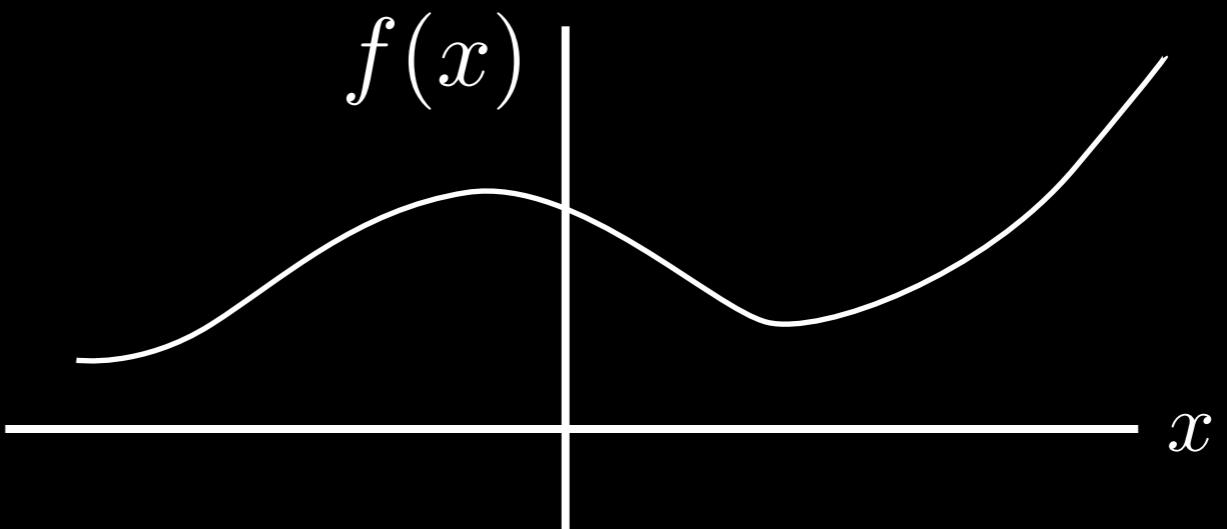
$$f(x, y) = \sin(2^x x)$$

frequency increase with x

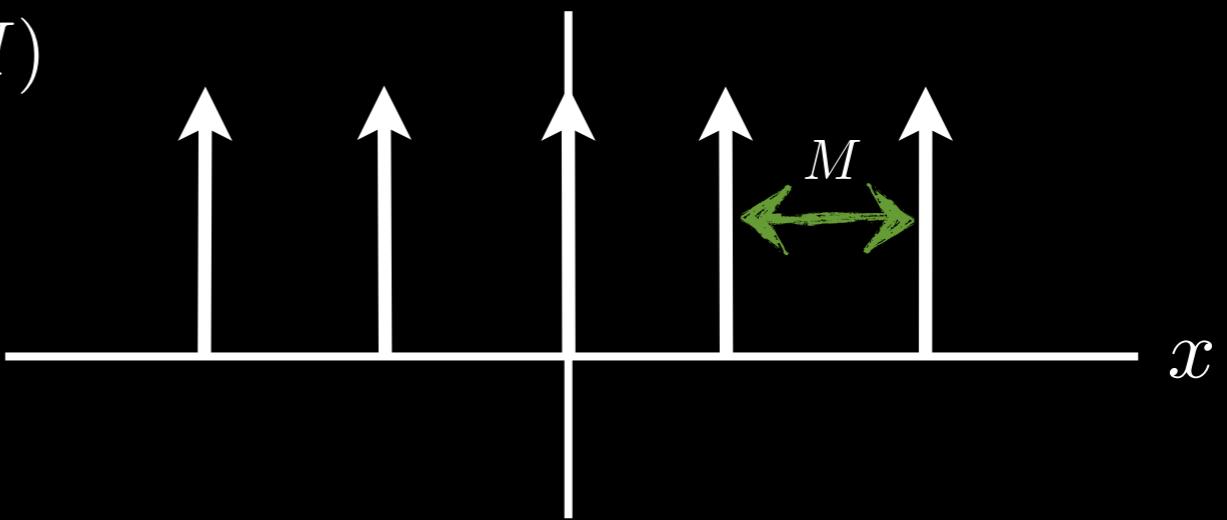
ALIASING
not enough samples

Aliasing as seen in
the frequency domain

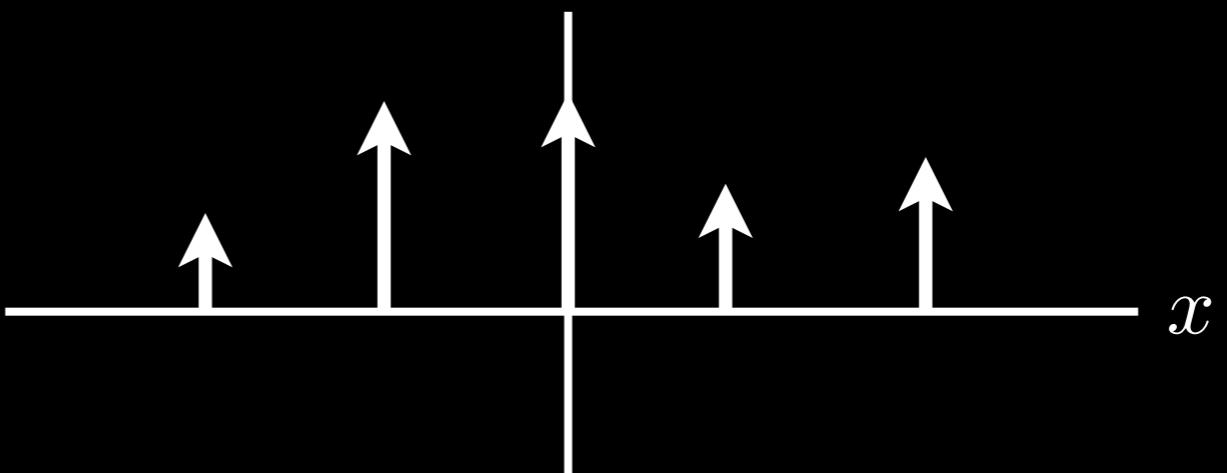
spatial
domain



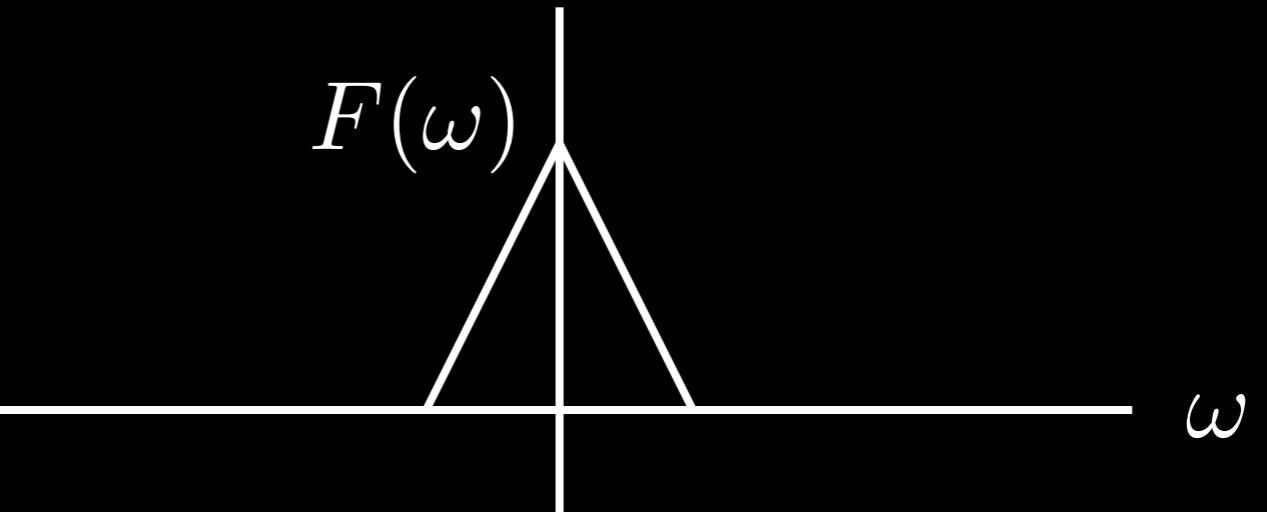
$$\text{comb}(x) = \sum_{k=-\infty}^{\infty} \delta(x - kM)$$



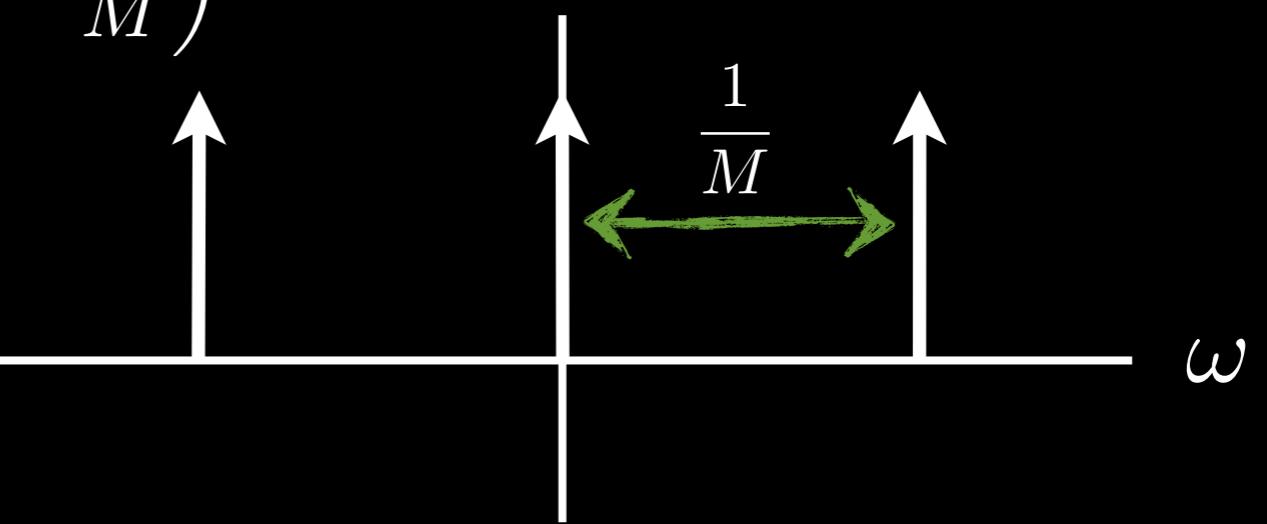
multiply



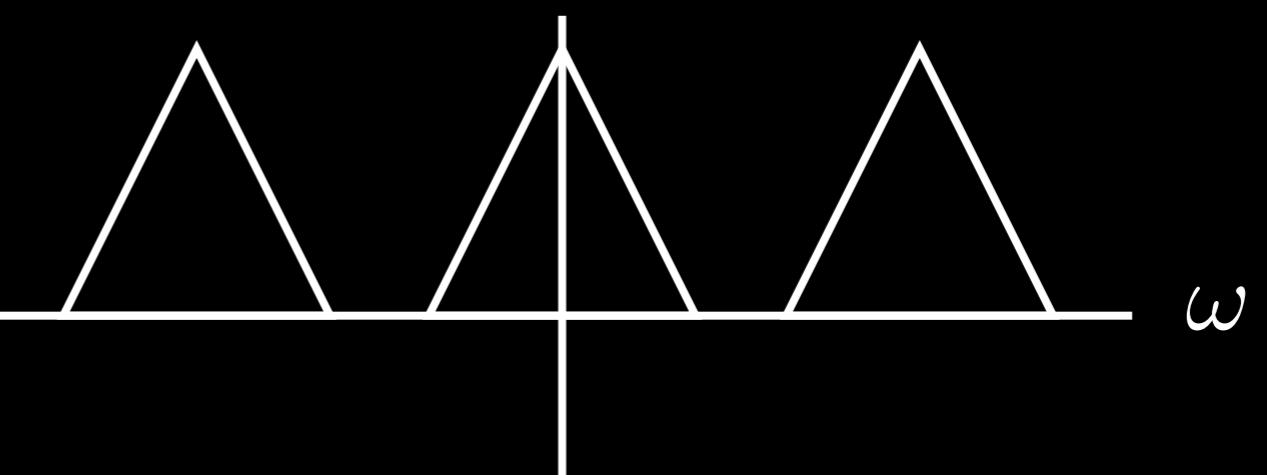
frequency
domain

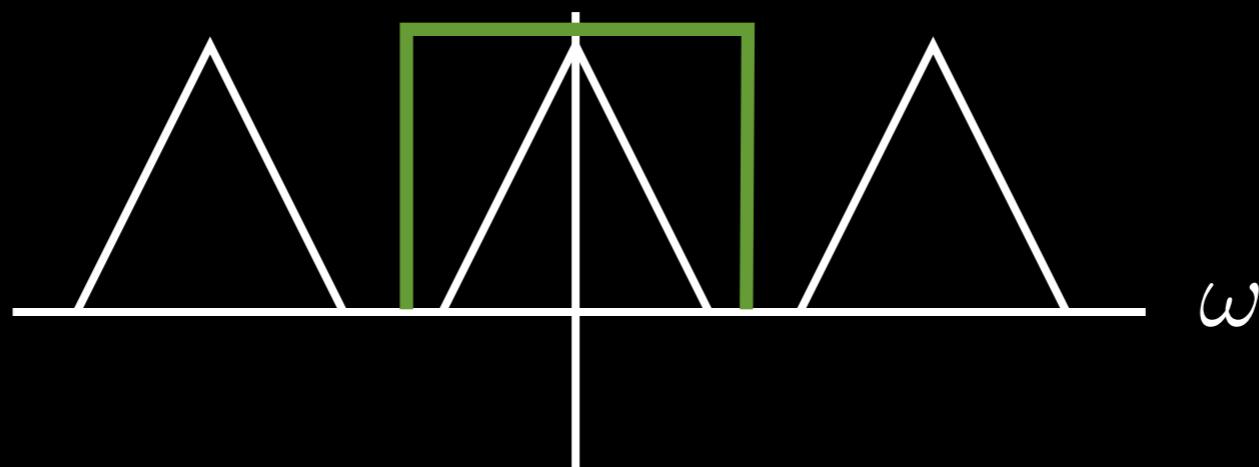


$$\text{comb}(\omega) = \frac{1}{M} \sum_{k=-\infty}^{\infty} \delta\left(\omega - k \frac{1}{M}\right)$$

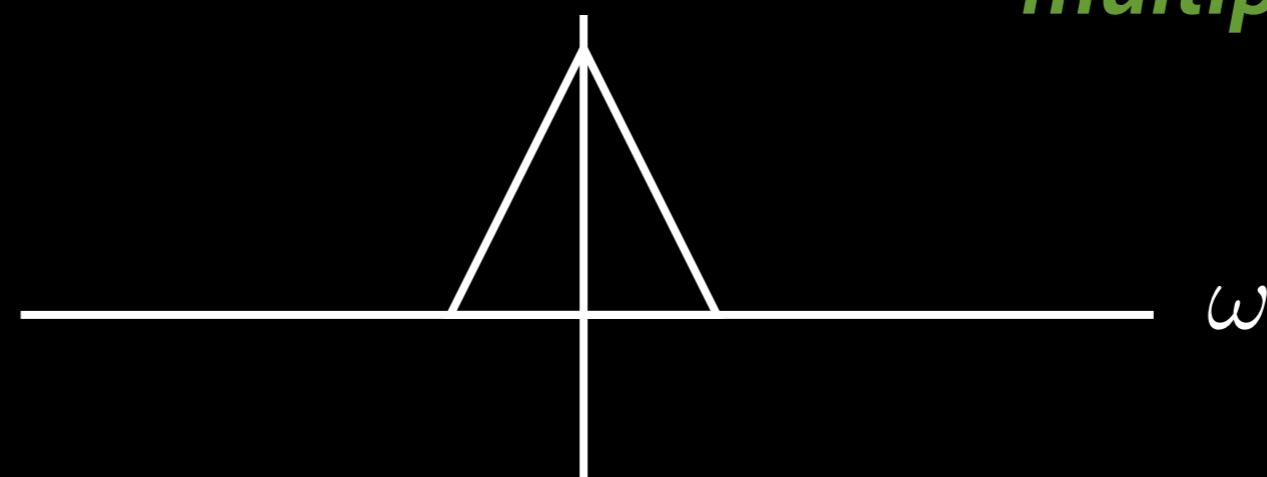


convolve

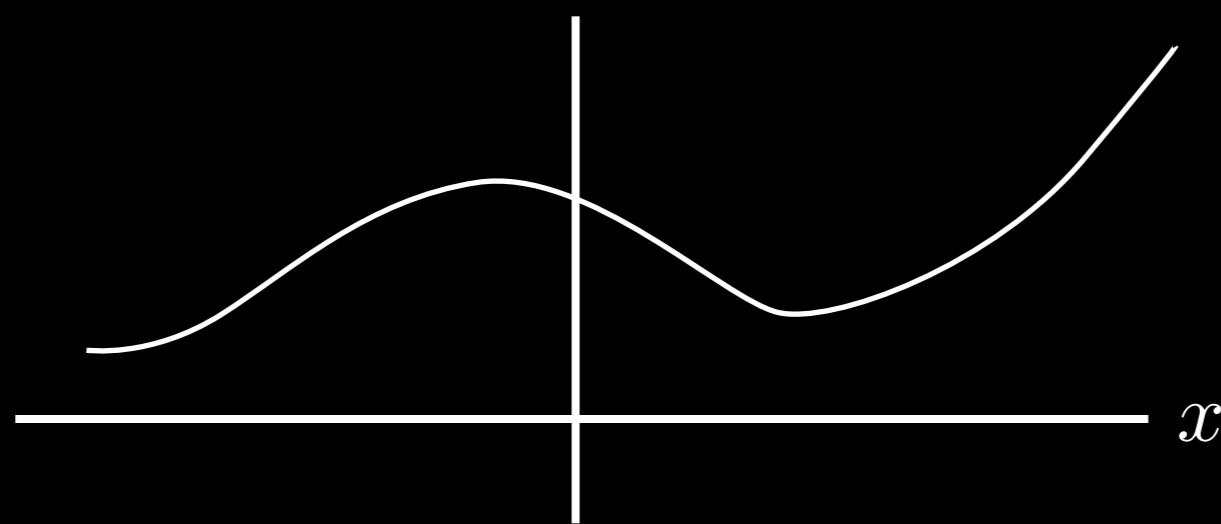




multiply by box filter



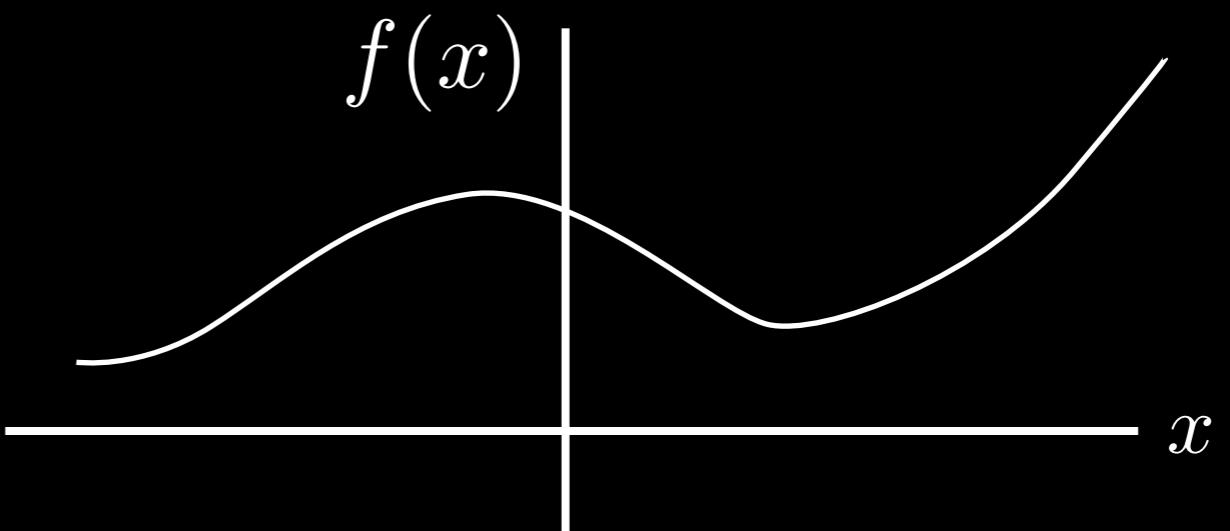
inverse Fourier transform



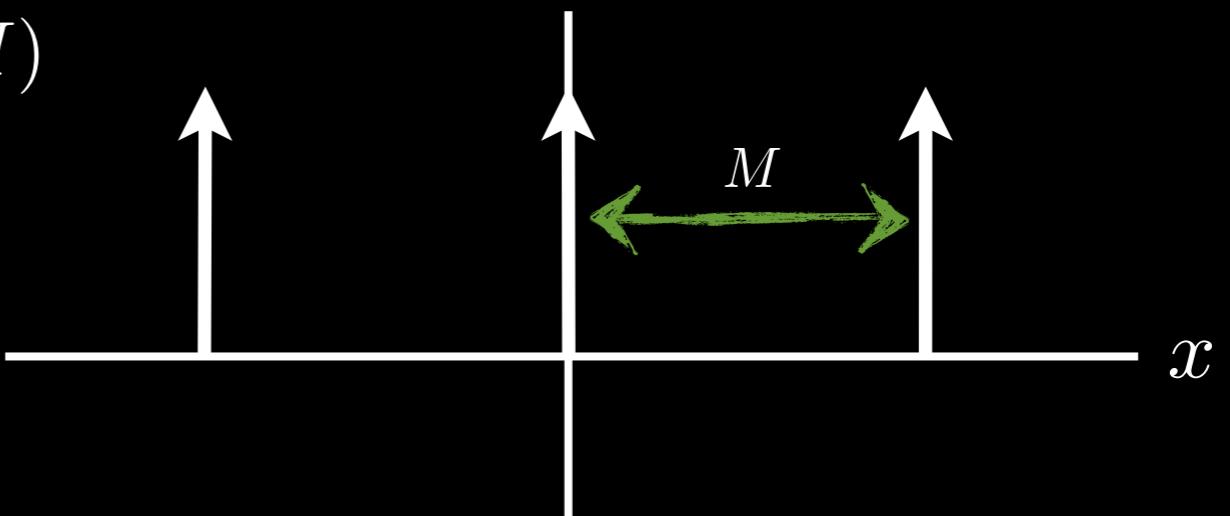
$$f(x) = \left(\sum_{k=-\infty}^{\infty} f(kM) \delta(x - kM) \right) * \text{sinc}\left(\frac{\pi x}{M}\right)$$

$$= \sum_{k=-\infty}^{\infty} f(kM) \text{sinc}\left(\frac{\pi}{M}(x - kM)\right)$$

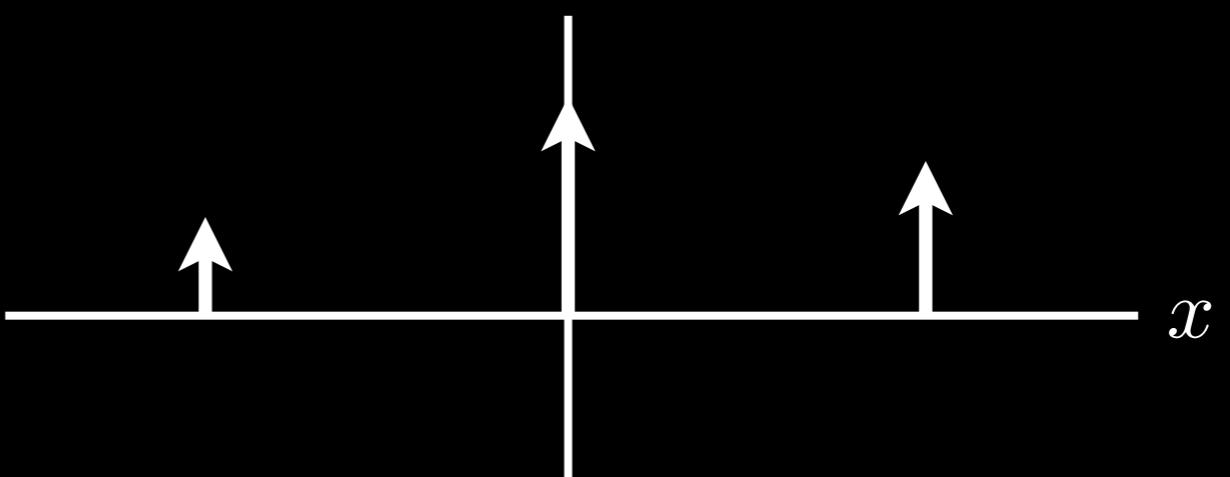
spatial
domain



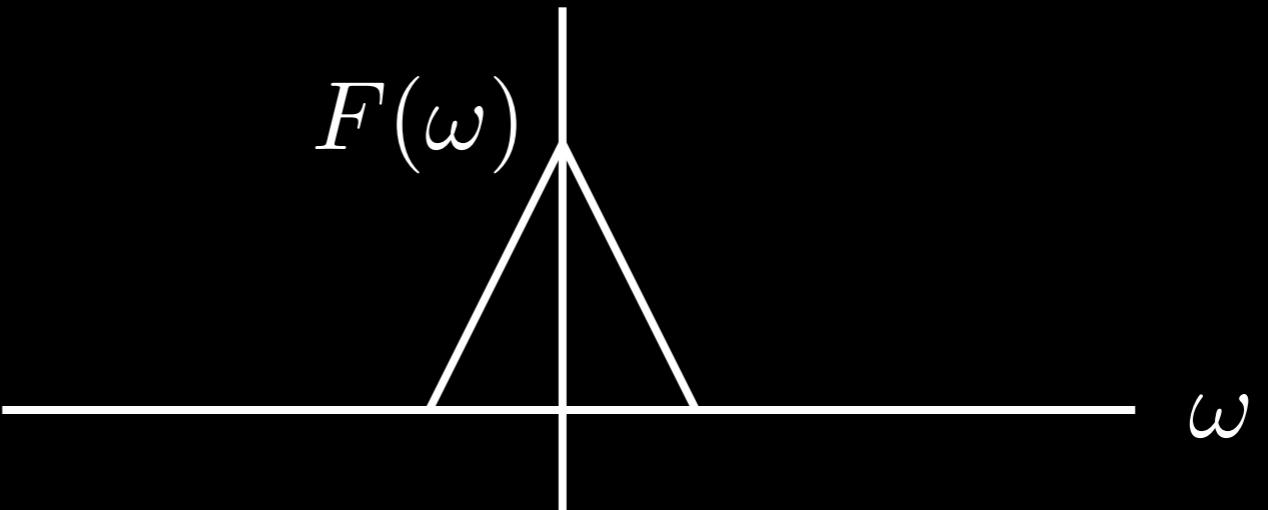
$$\text{comb}(x) = \sum_{k=-\infty}^{\infty} \delta(x - kM)$$



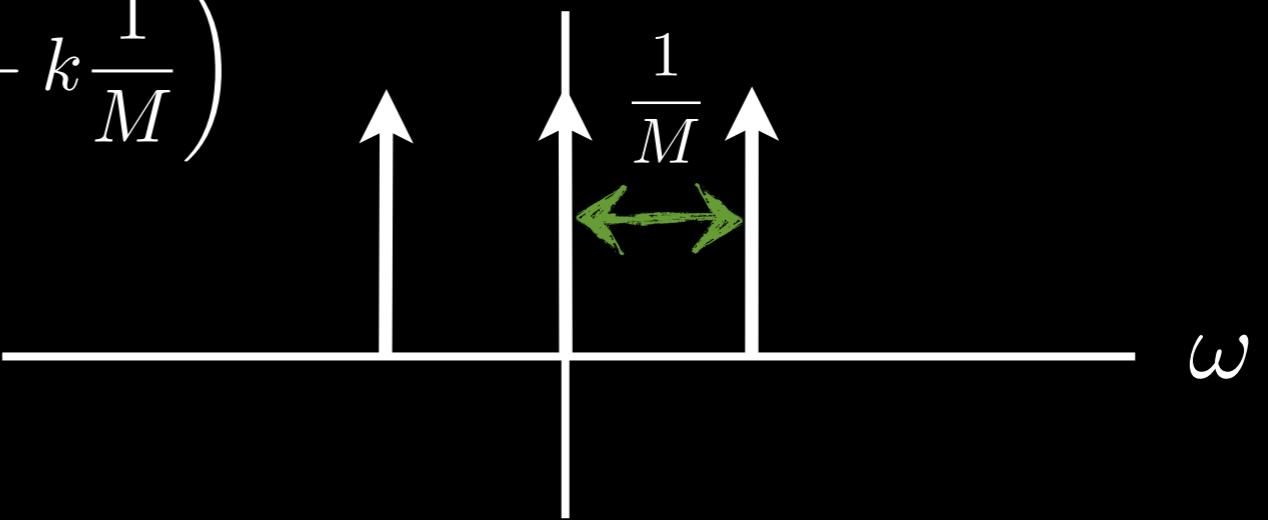
multiply



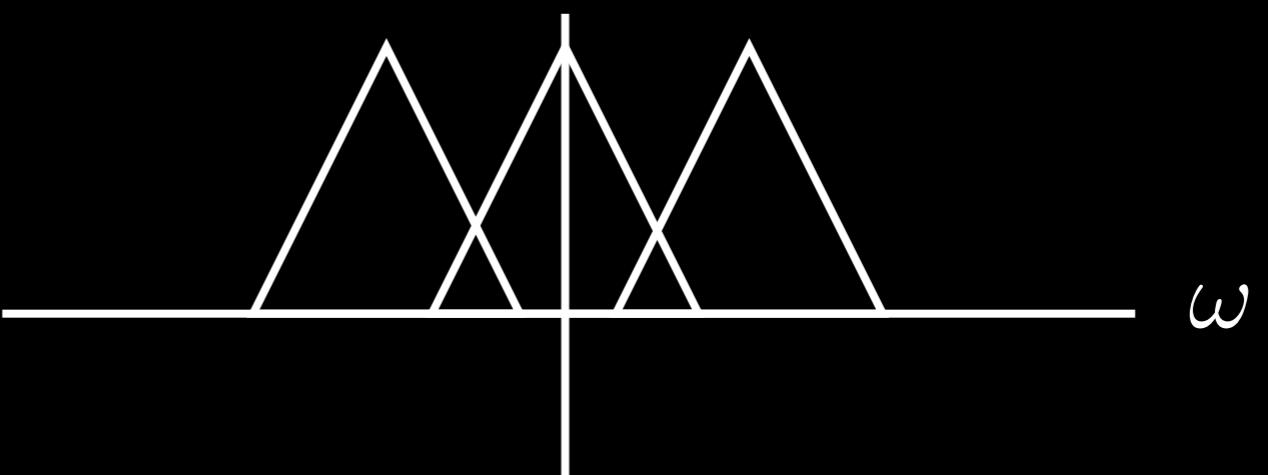
frequency
domain

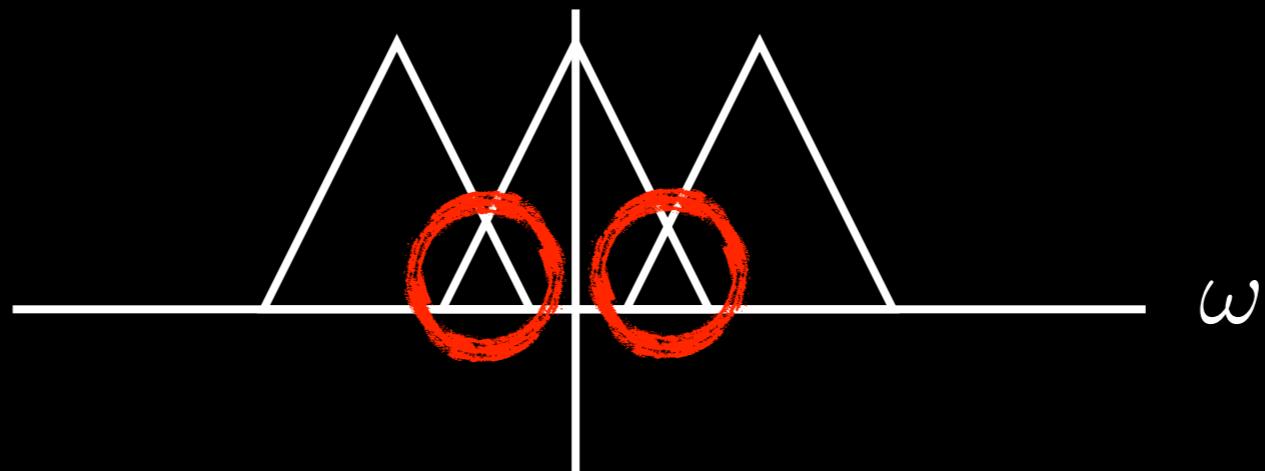


$$\text{comb}(\omega) = \frac{1}{M} \sum_{k=-\infty}^{\infty} \delta \left(\omega - k \frac{1}{M} \right)$$



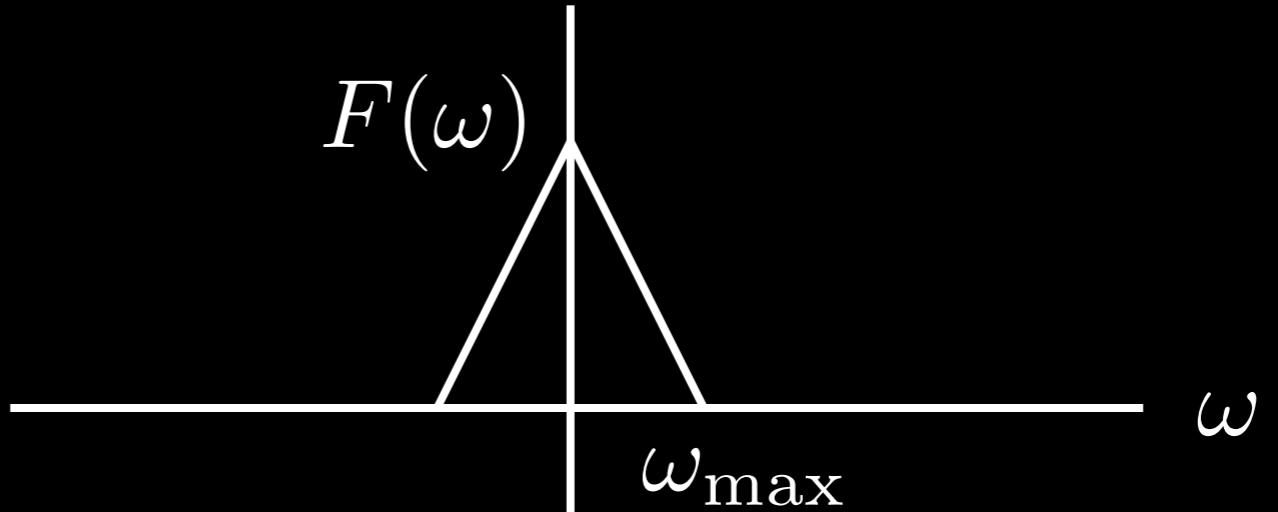
convolve



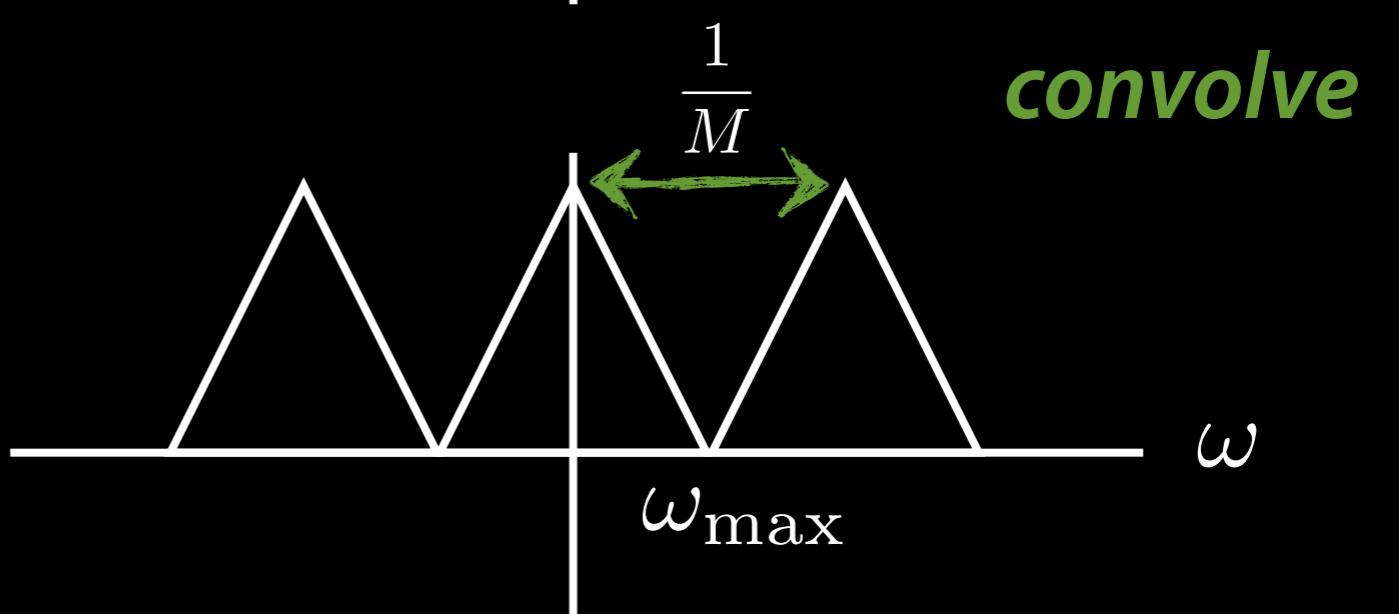
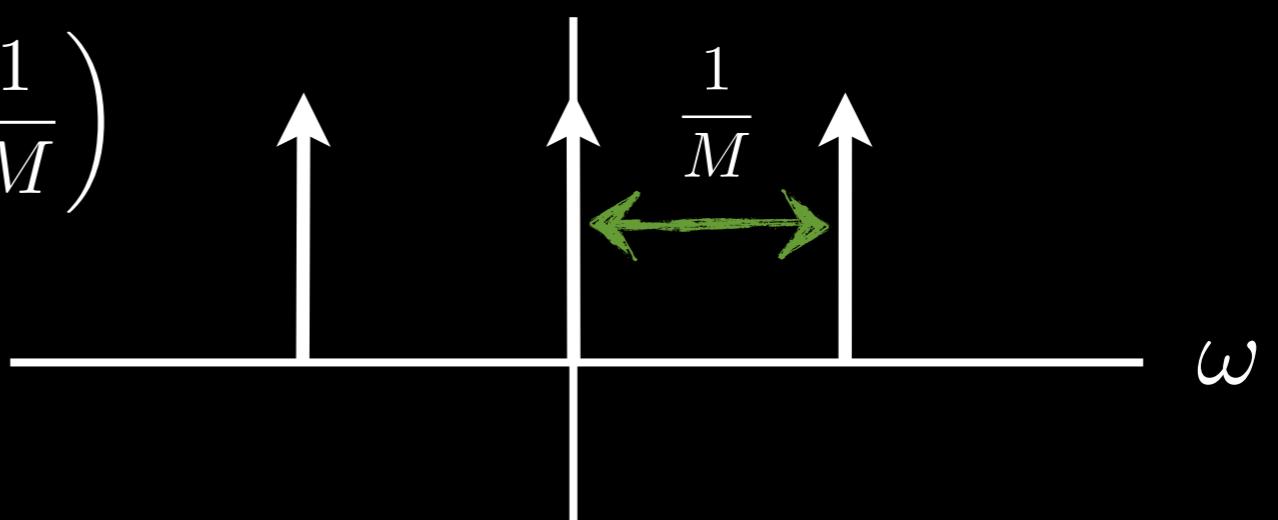


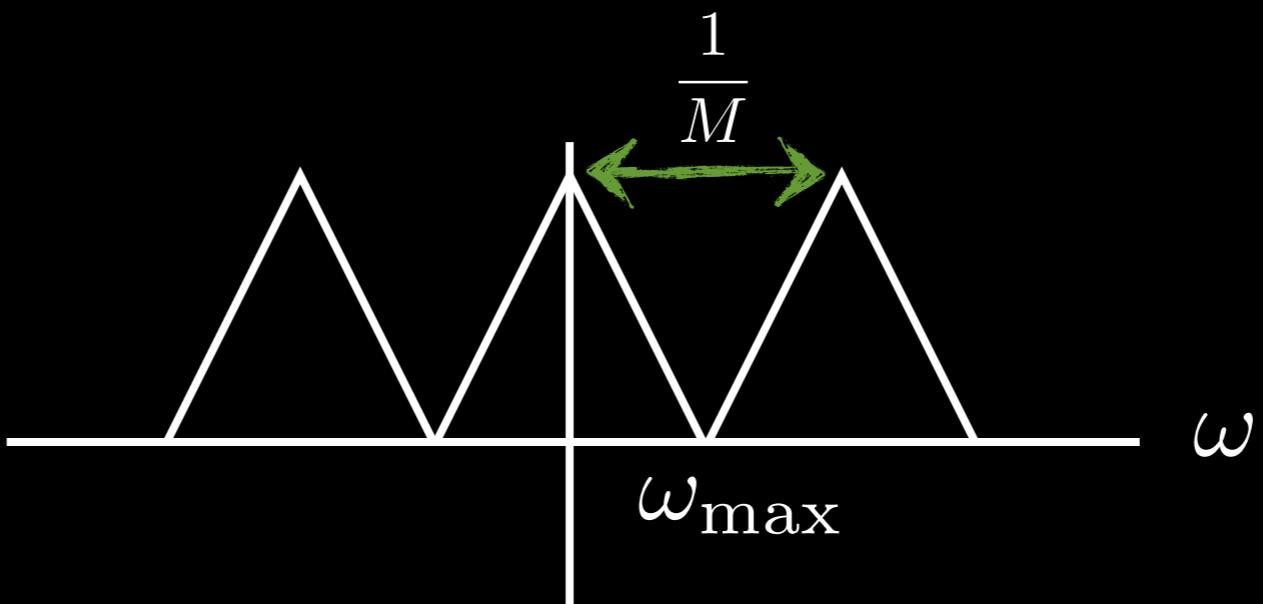
Signal is **ALIASED**

Nyquist Sampling Theorem

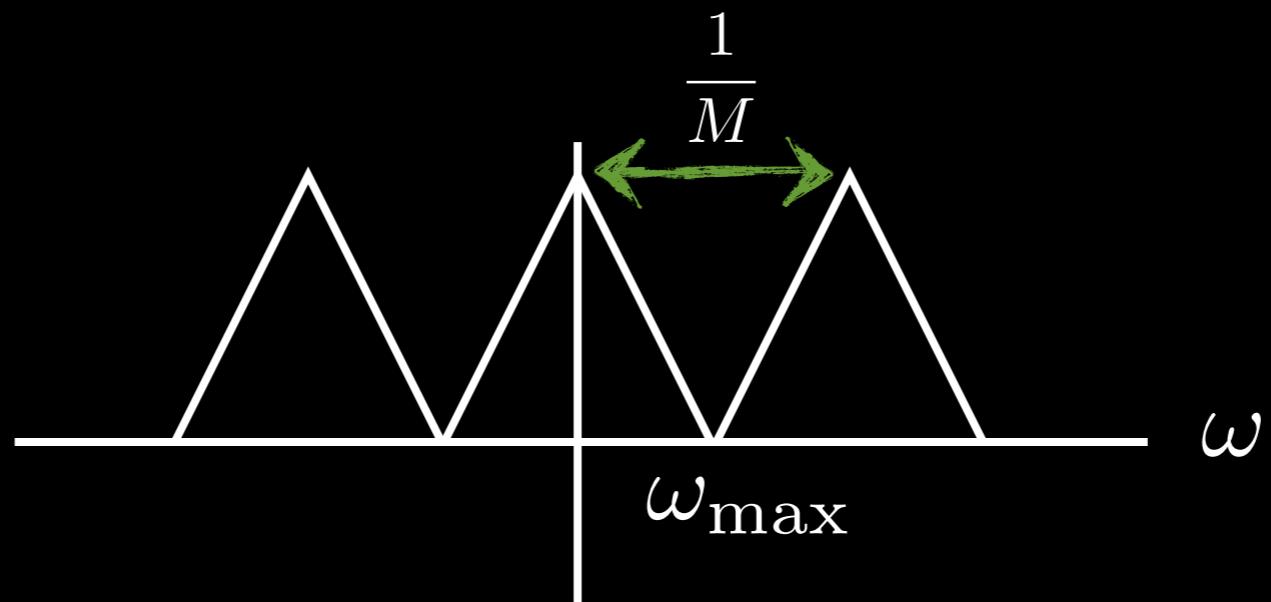


$$\text{comb}(\omega) = \frac{1}{M} \sum_{k=-\infty}^{\infty} \delta\left(\omega - k \frac{1}{M}\right)$$



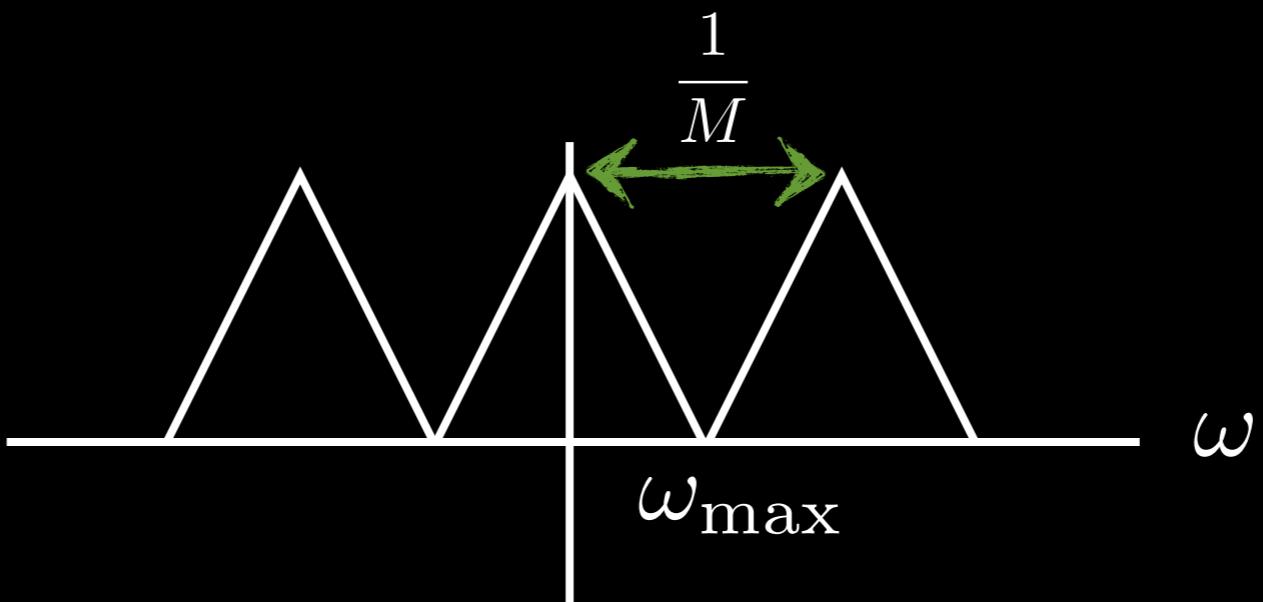


When does **aliasing** not occur?



Nyquist Sampling Theorem

$$\frac{1}{M} > 2\omega_{\text{max}}$$



Nyquist Sampling Theorem

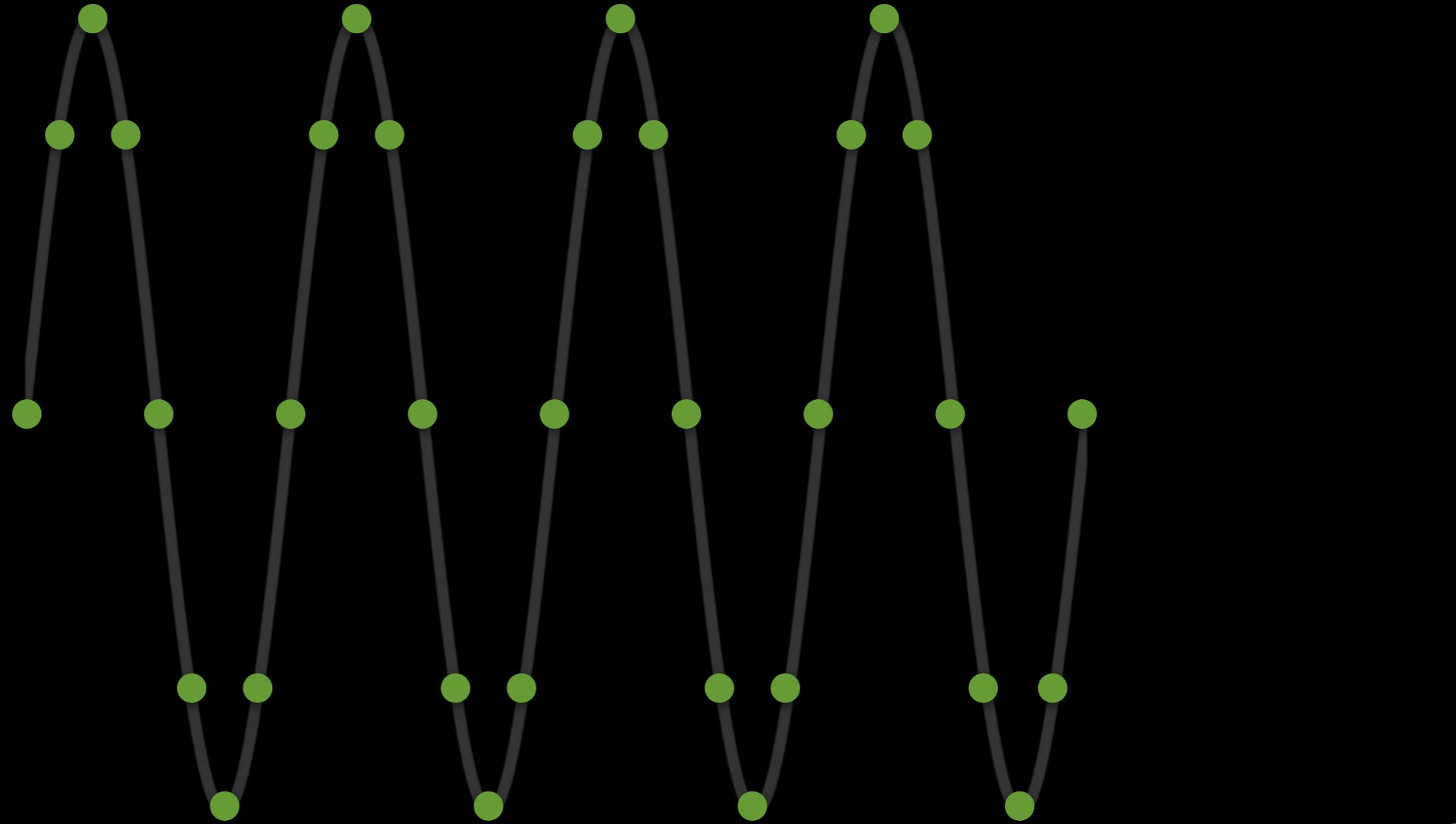
$$\frac{1}{M} > 2\omega_{\max}$$

Sampling frequency must be greater than twice the signal's highest frequency

How can we avoid
aliasing?

1

Adjust sampling rate



Sample signal more often

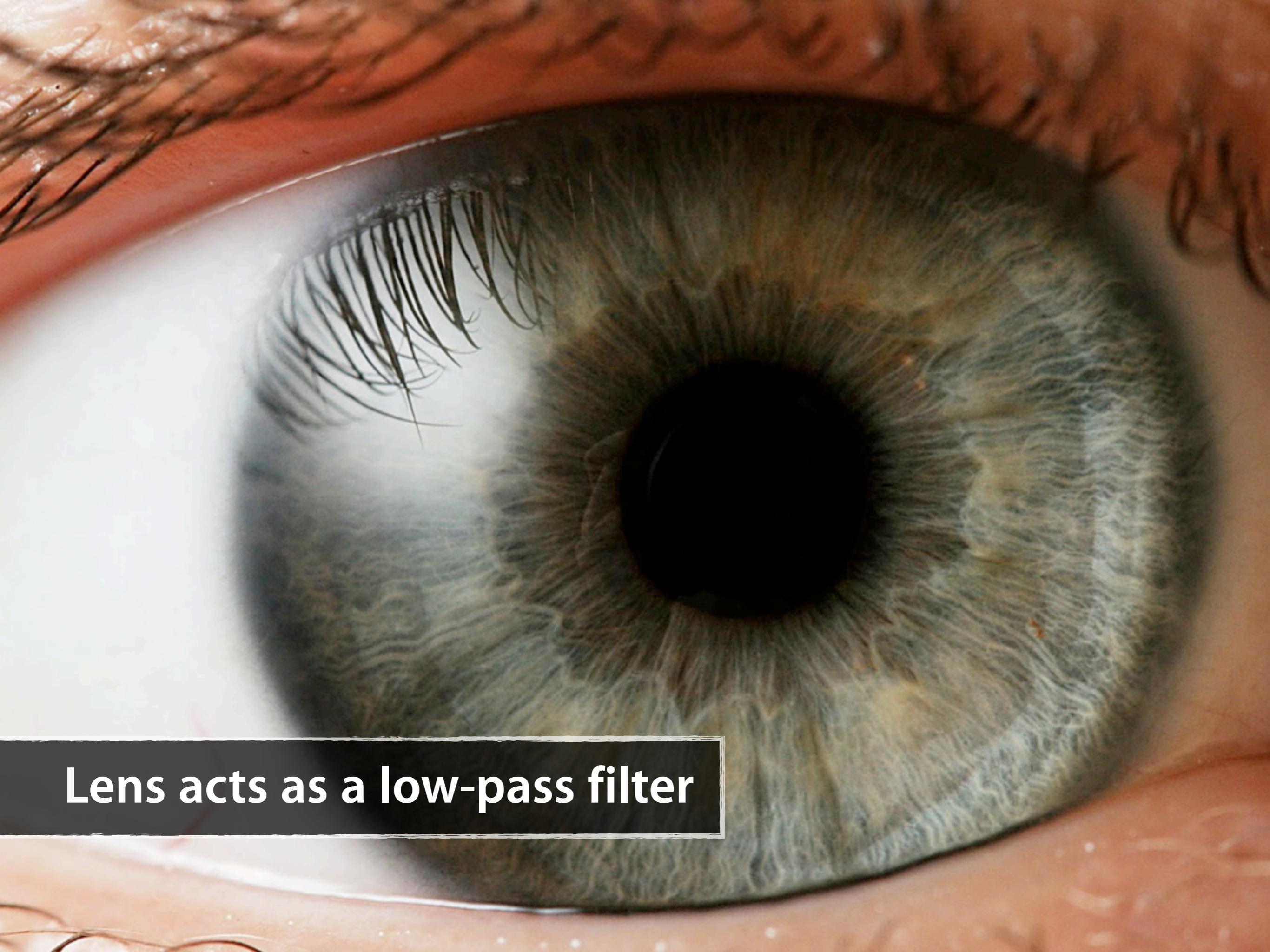
2

Prefilter image

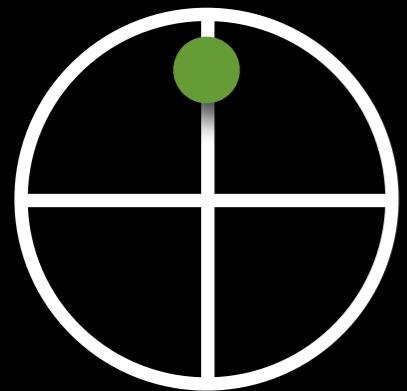
Remove some

HIGH FREQUENCIES

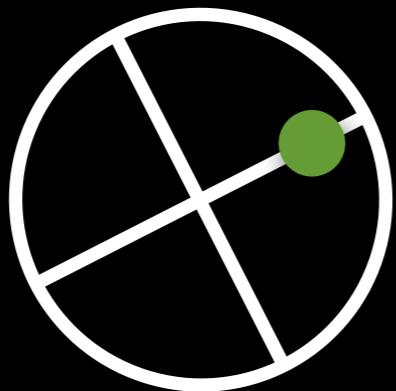
smooth prior to sampling



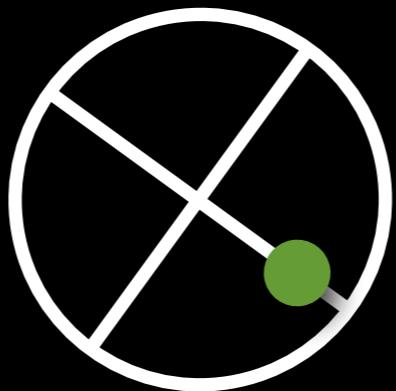
Lens acts as a low-pass filter



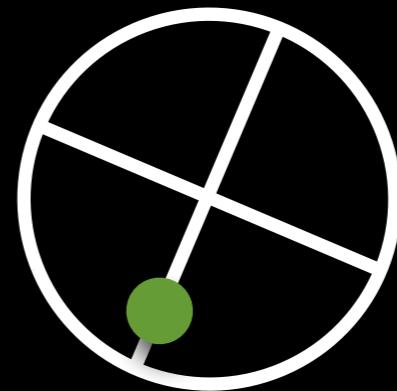
frame 0



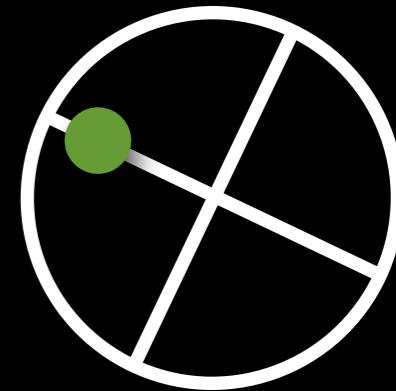
frame 1



frame 2



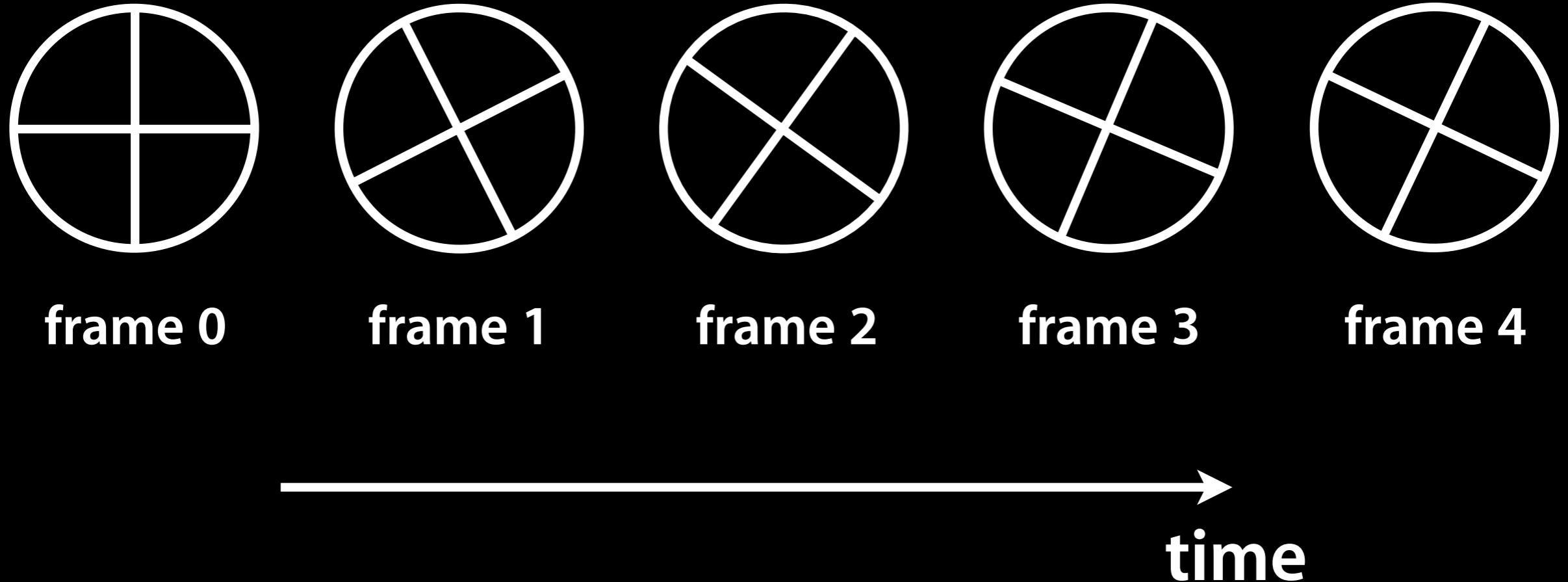
frame 3



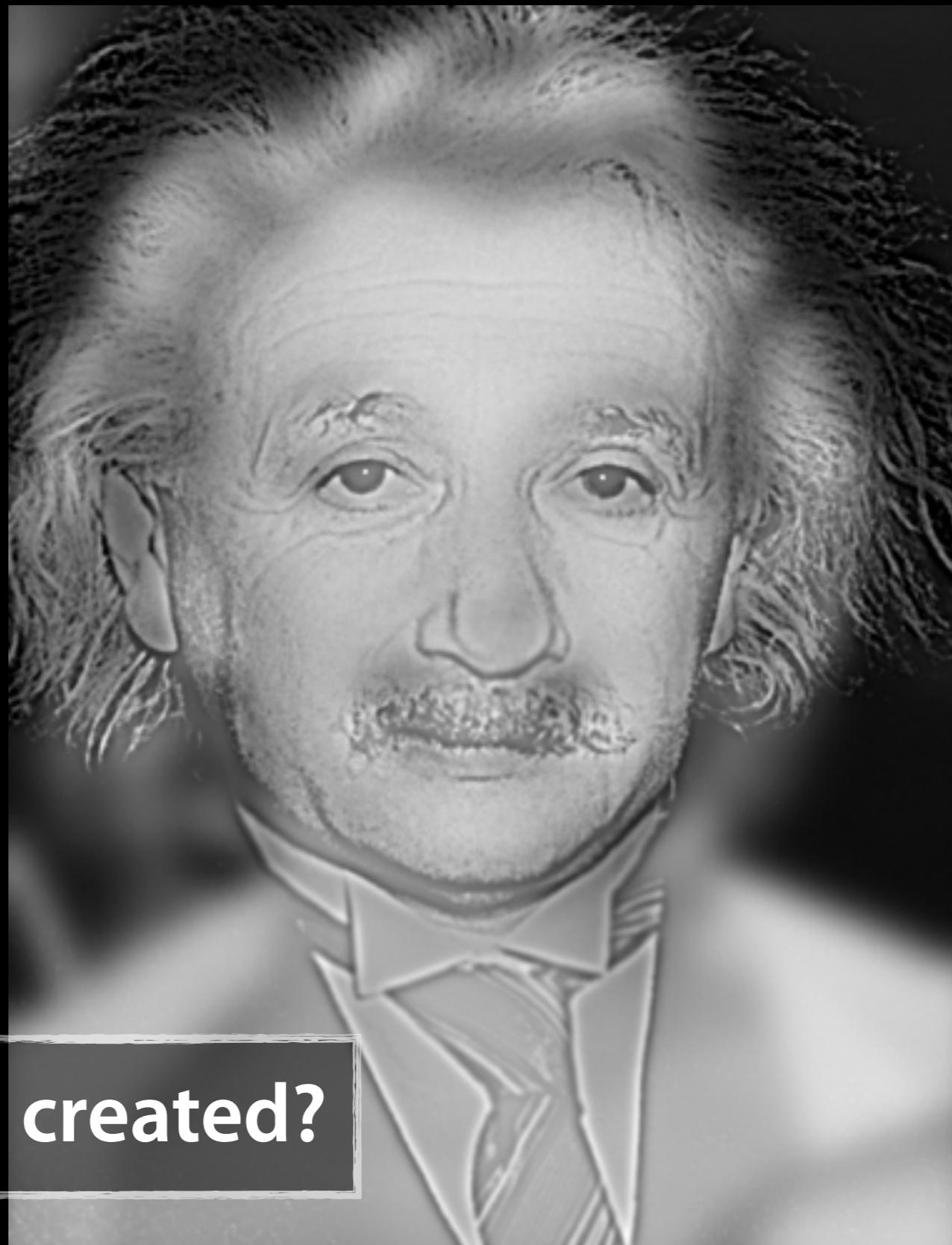
frame 4



time



Without the dot wheel appears to slowly rotate backwards!



How was this created?

distributive

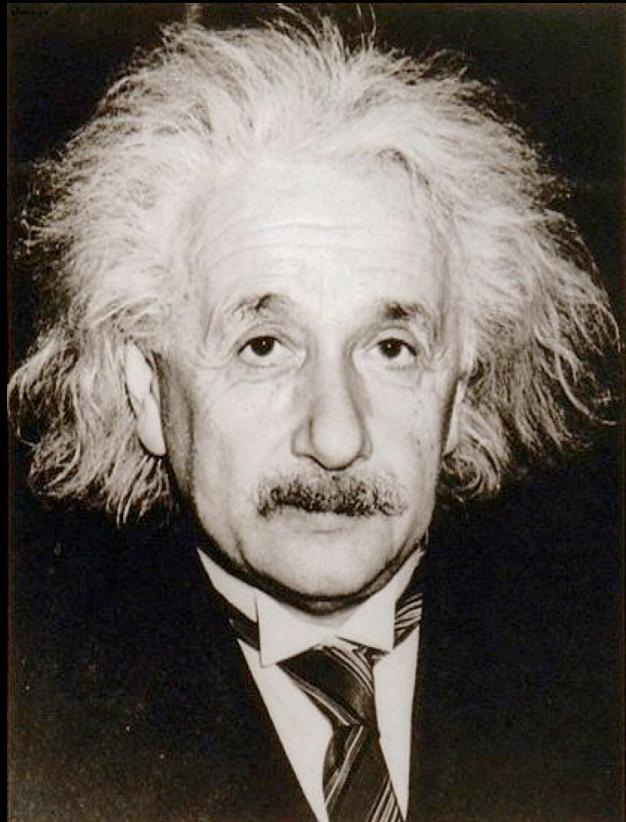
$$E[x, y] * (F[x, y] + G[x, y])$$

$$= (E[x, y] * F[x, y]) + (E[x, y] * G[x, y])$$



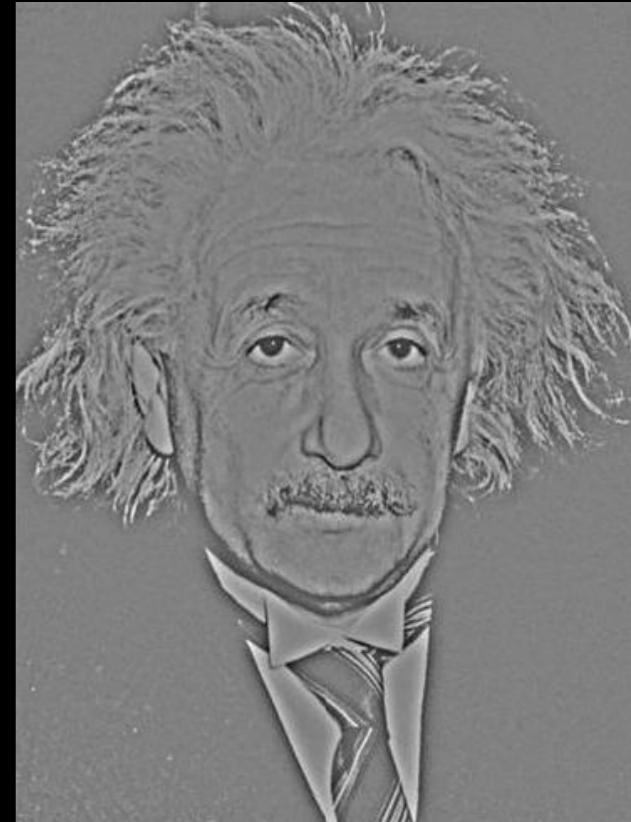
$$* \quad g_1[x, y] =$$

**lowpass
filter**

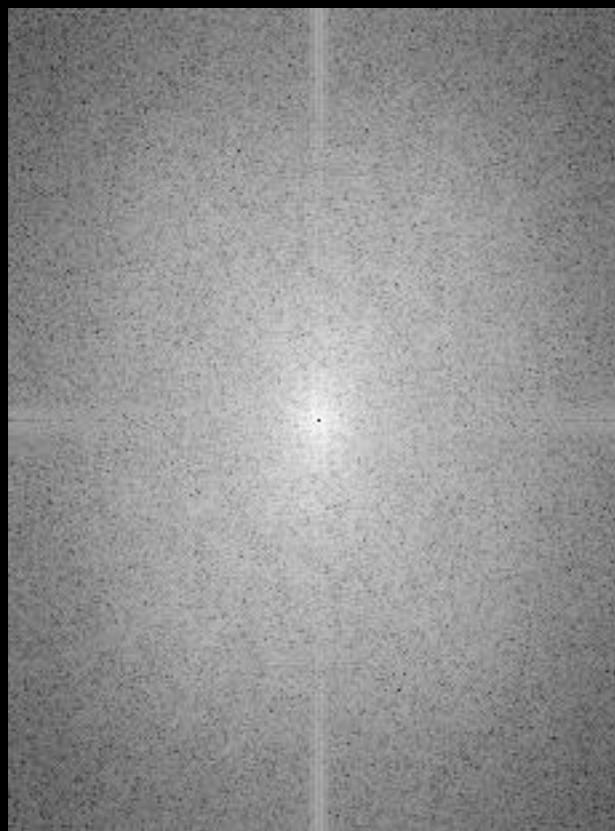


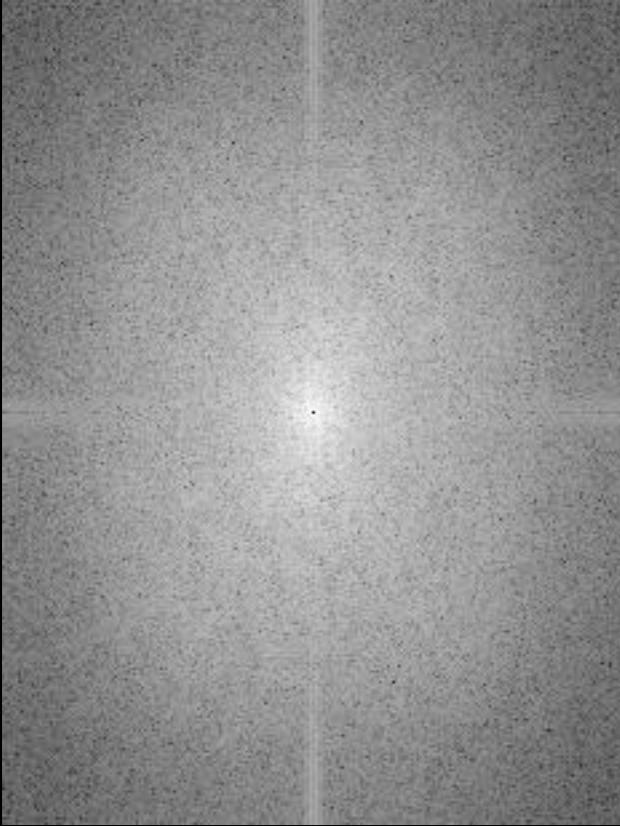
$$* (\delta[x, y] - g_2[x, y]) =$$

**highpass
filter**

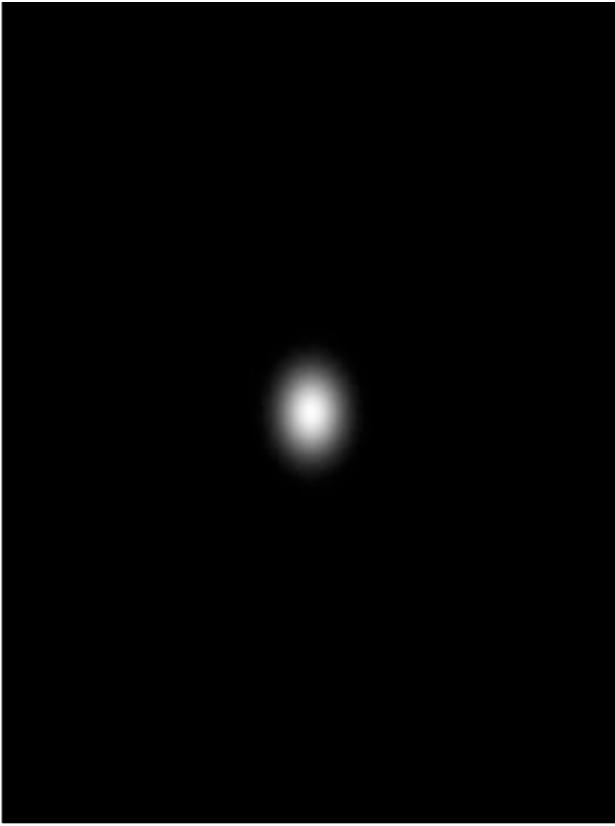


Frequency
Domain

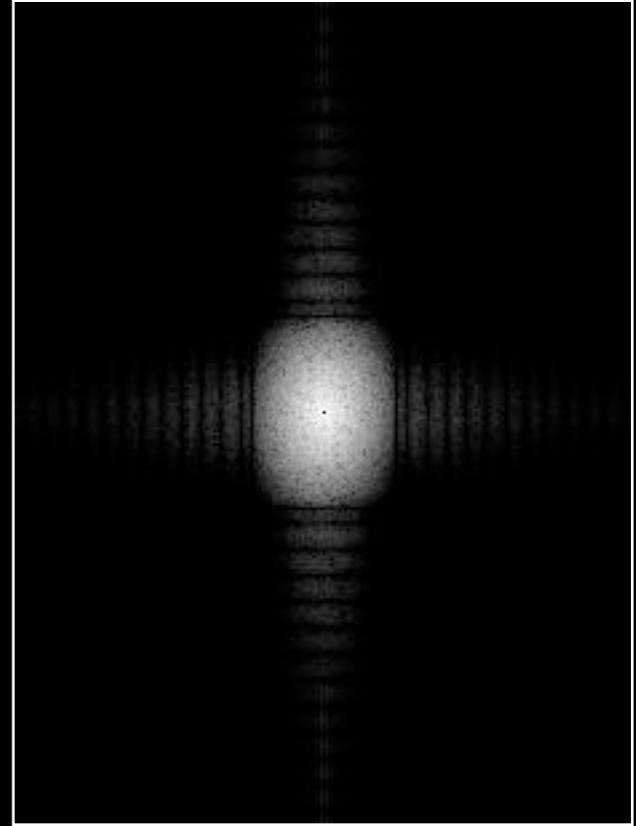




\times



$=$



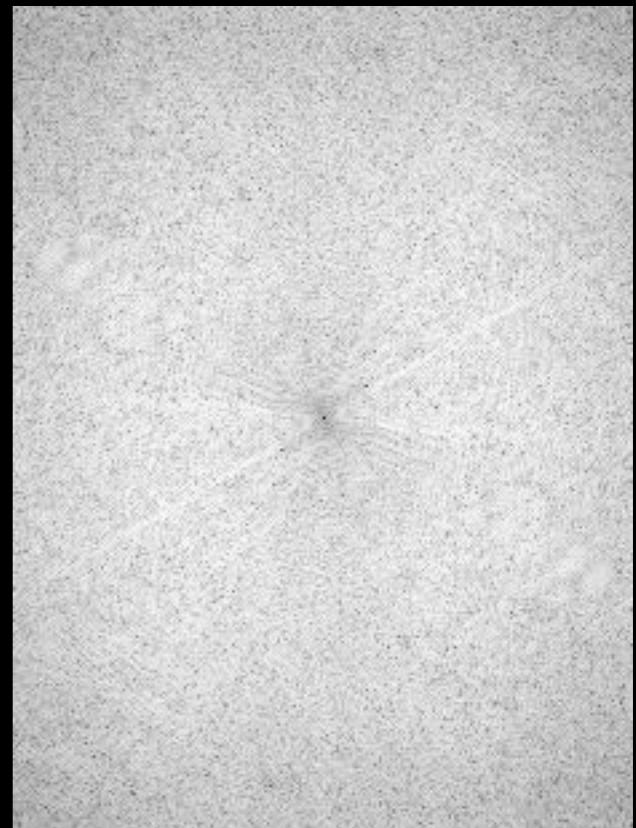
$+$



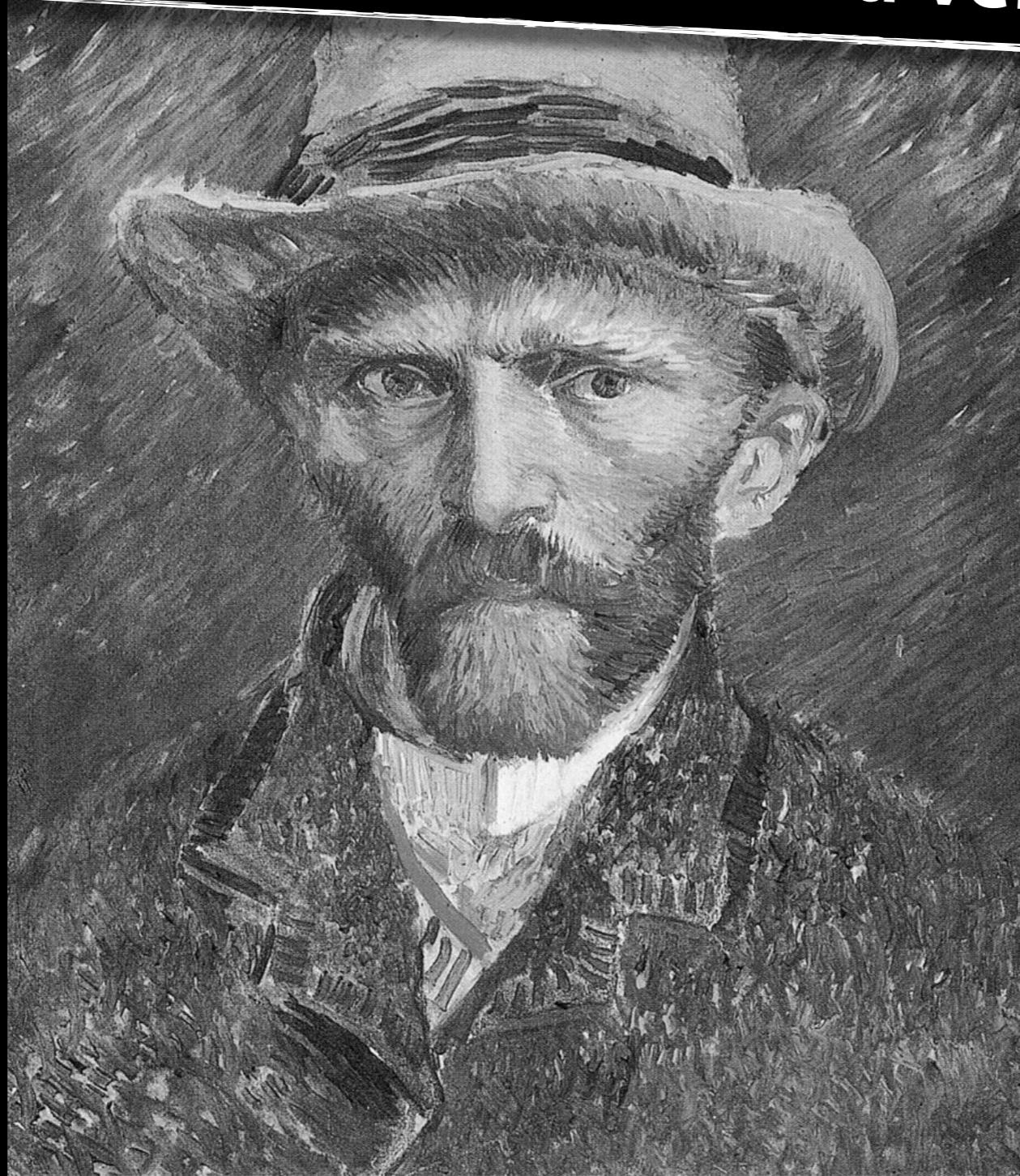
\times



$=$



How to generate a half-sized version?





1/4 size (4x zoom)

Smooth then subsample



original



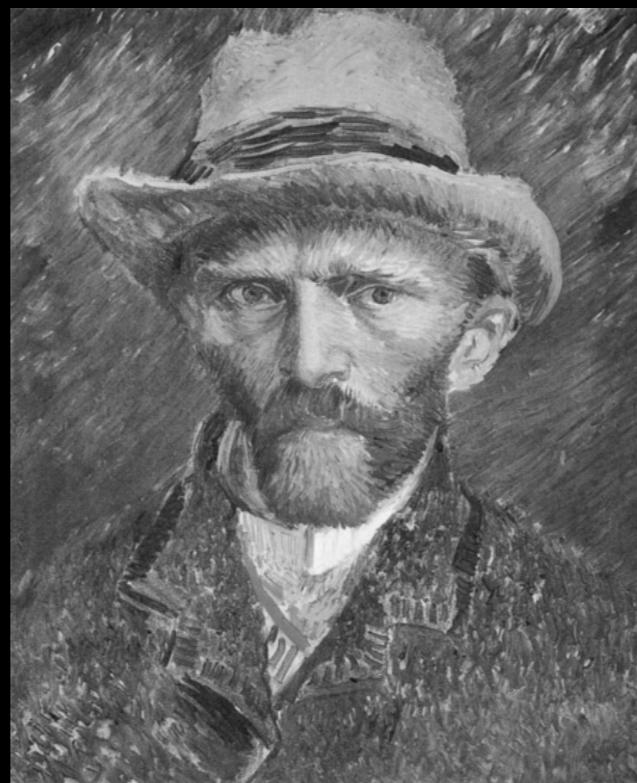
binomial
repeated smoothing and subsampling

Gaussian
Pyramid



binomial
✓
repeated smoothing and subsampling

Gaussian Pyramid



1/2 size

```
>> f = (1/16)*[1 4 6 4 1]; % Binomial filter
>> tmp = imfilter(im, f, 'symmetric');
>> im_smooth = imfilter(tmp, f', 'symmetric');
>> im_smooth = im_smooth(1:2:end, 1:2:end);
>> imshow(im_smooth, [])
```

Gaussian Pyramid



1/2 size

```
>> f = (1/16)*[1 4 6 4 1]; % Binomial filter
>> tmp = imfilter(im, f, 'symmetric');
>> im_smooth = imfilter(tmp, f', 'symmetric');
>> im_smooth = im_smooth(1:2:end, 1:2:end);
>> imshow(im_smooth, [])
```

Pascal's
Triangle

		1			
		1	1		
	1	2	1		
1	3	3	1		
1	4	6	4	1	
1	5	10	10	5	1

Gaussian Pyramid



1/2 size

```
>> f = (1/16)*[1 4 6 4 1]; % Binomial filter
>> tmp = imfilter(im, f, 'symmetric');
>> im_smooth = imfilter(tmp, f', 'symmetric');
>> im_smooth = im_smooth(1:2:end);
>> imshow(im_smooth, [])
```

separable filtering



BLOCKY ARTIFACTS due to compression

JPEG
Compression

5

major steps

Step 1

Convert colour space



Convert RGB image to YCrCb



Y



Cr



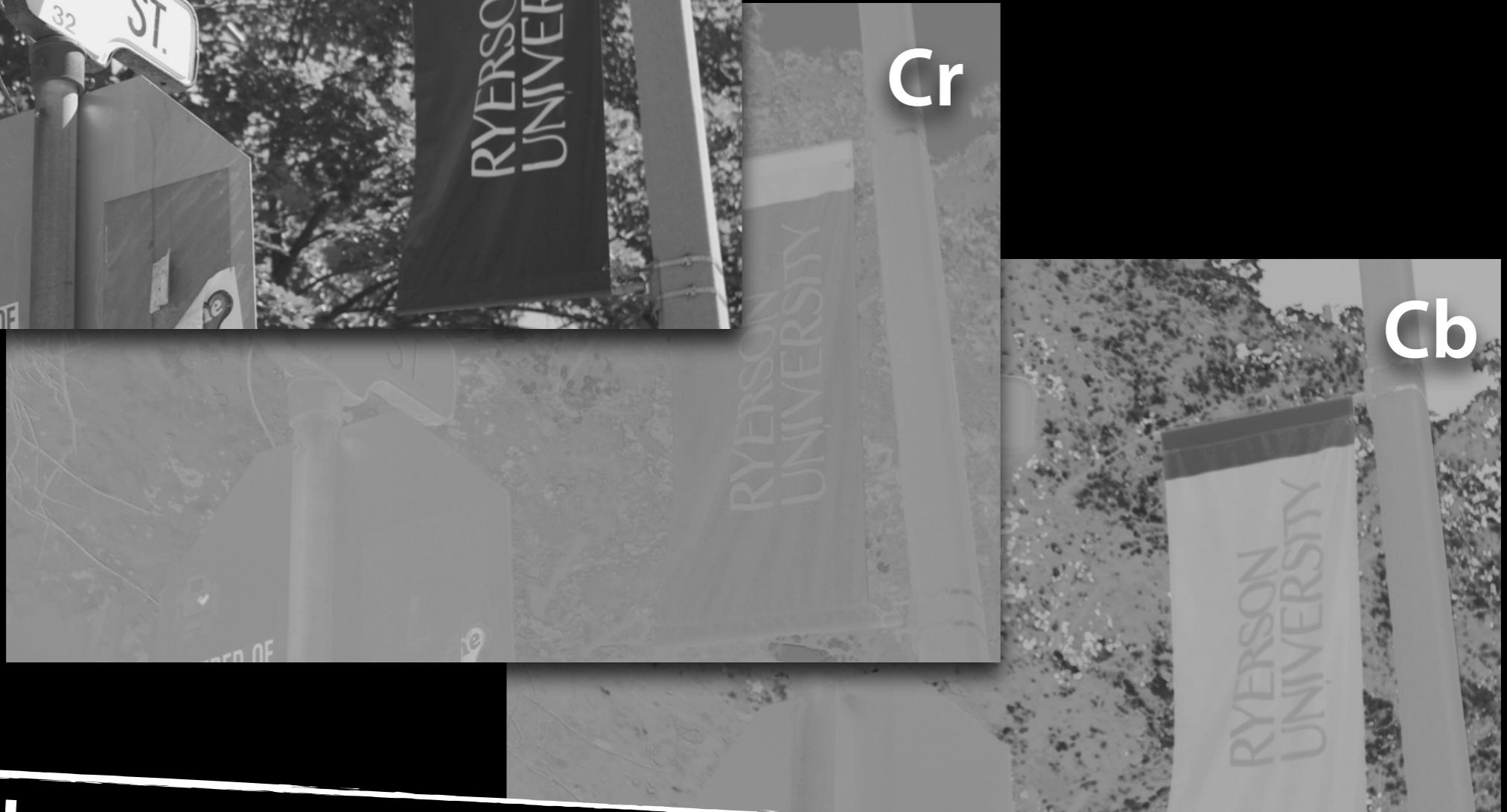
Cb

Step 2

Resample colour channels



Y



Cr

Cb

Subsample colour channels by factor of two



Cr



Cb

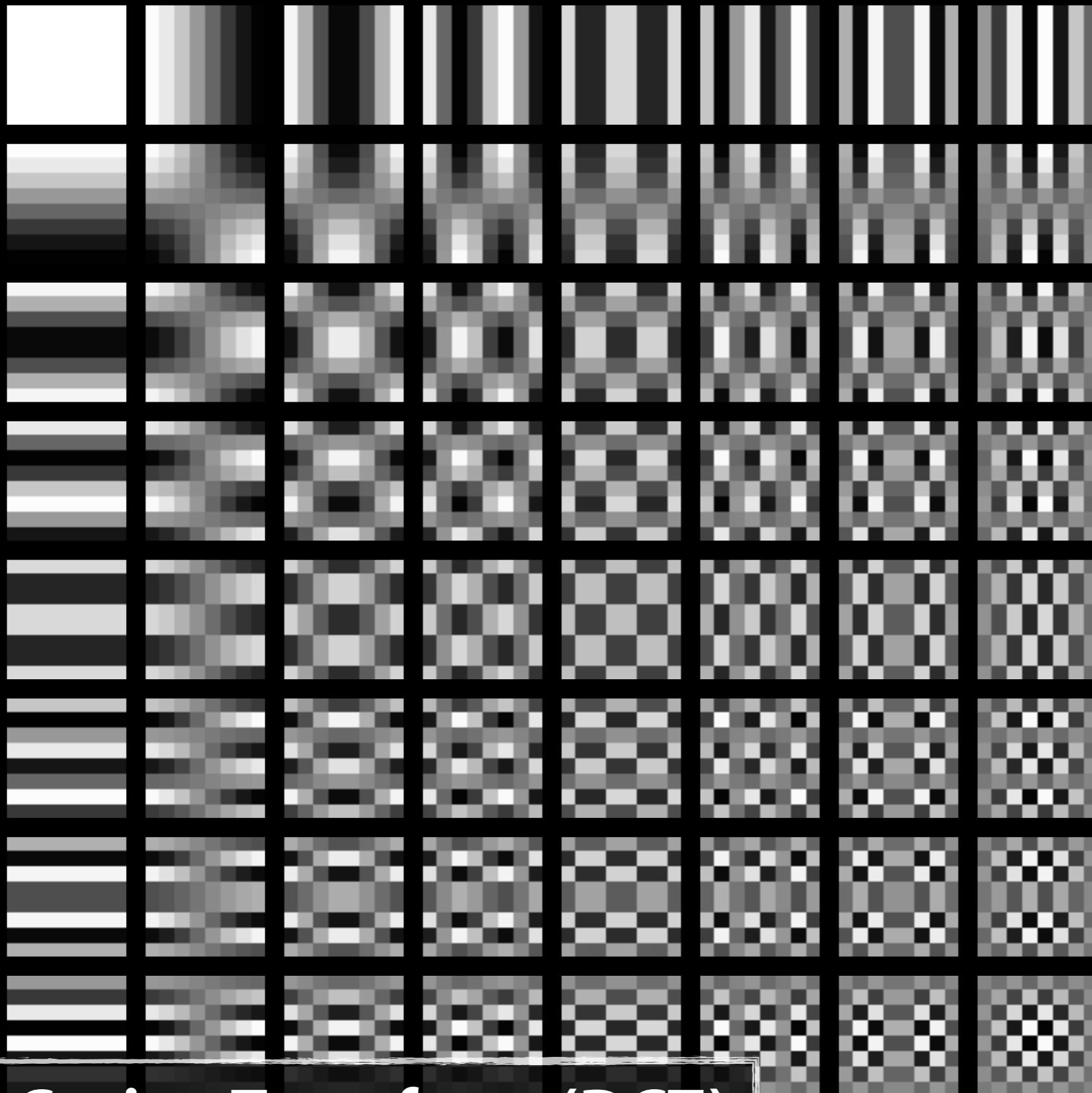


Y

Step 3

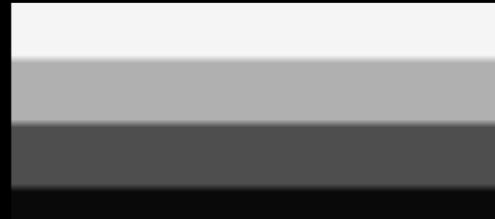
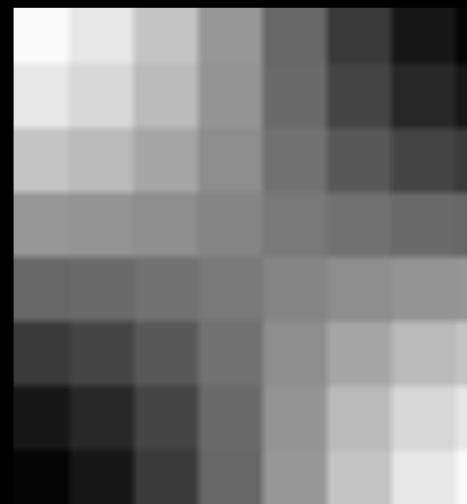
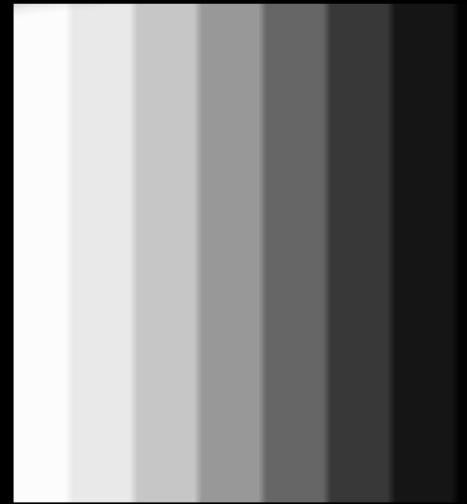
Compute DCT for each 8x8 image patch

map image patches to a set of real numbers

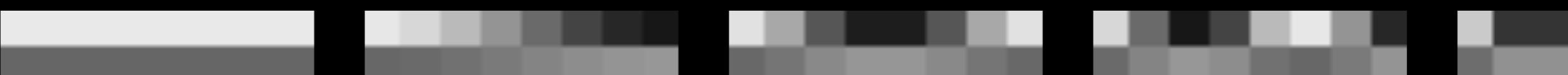
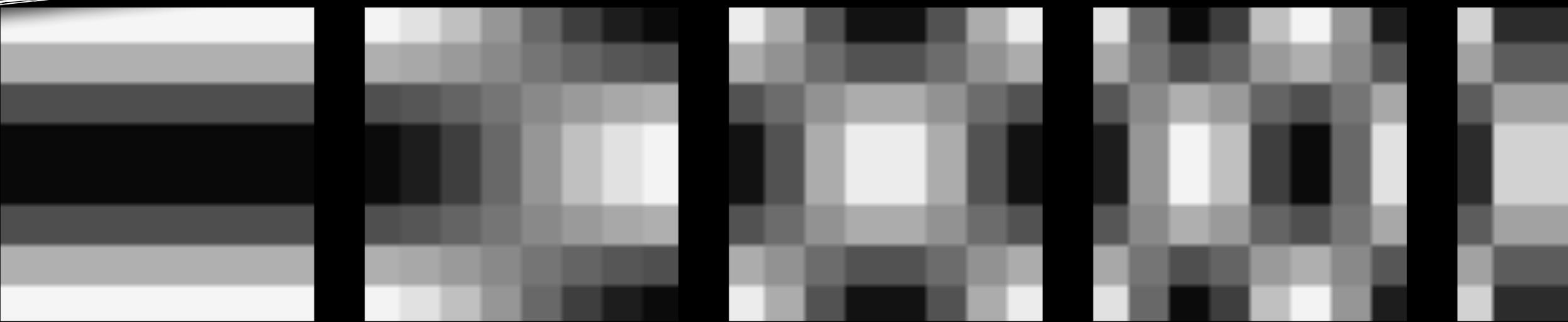


Discrete Cosine Transform (DCT)

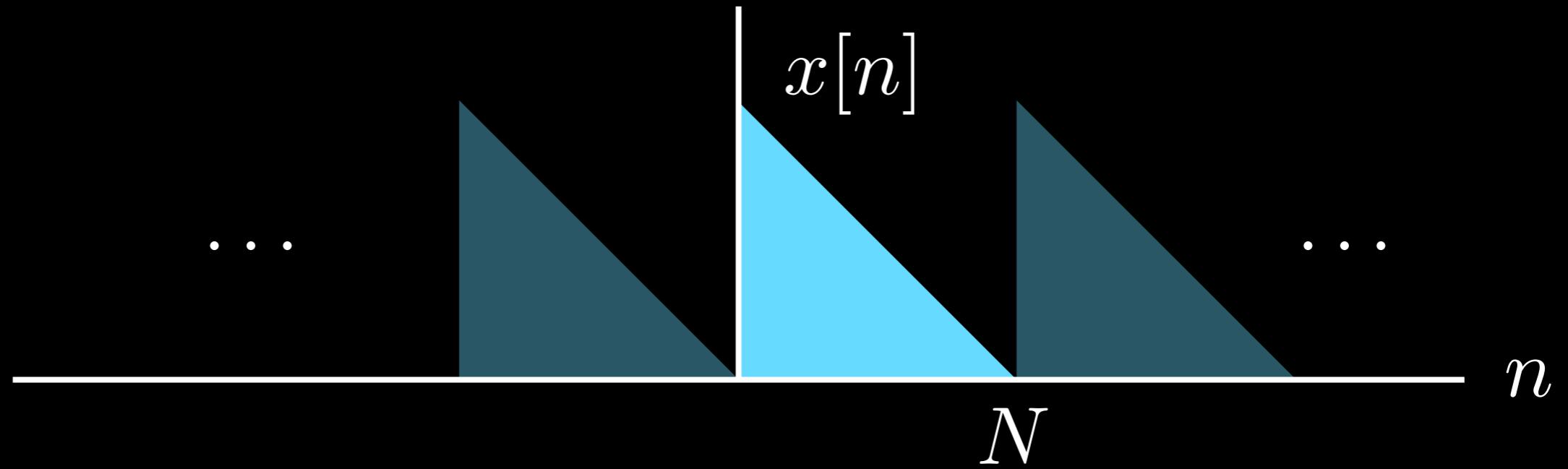
DC component capturing the average intensity



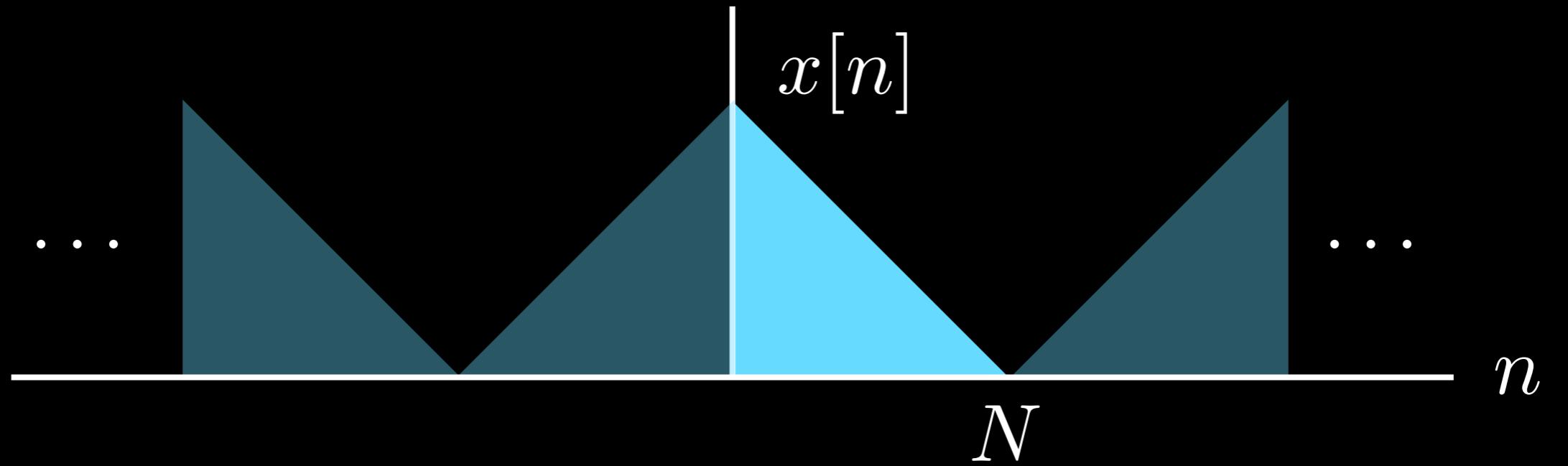
Basis images for the low frequencies



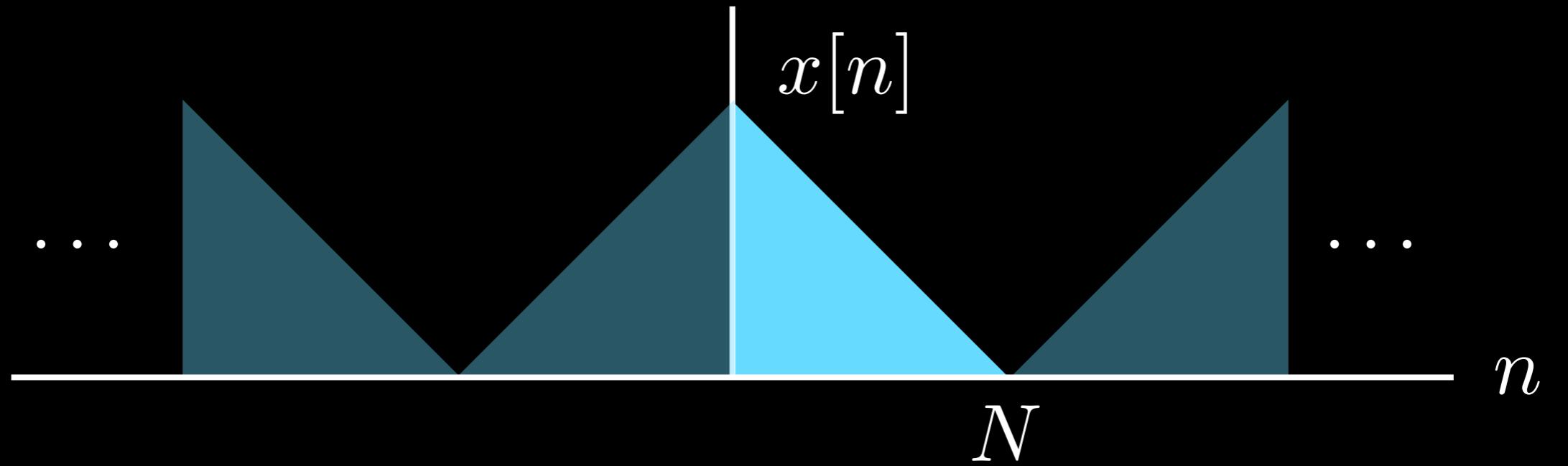
Basis images for the high frequencies



DFT assumes periodic extension beyond boundaries



DCT adds mirrored version and then periodic extension



Performing DFT without the complex numbers

Step 4

Quantize DCT coefficients

quantization level depends on frequency

Step 5

Lossless compression using Huffman coding

189

