3.2 Frequency Analysis

Outline

- Linear Shift-Invariant Systems
- The Fourier Transform
- The Wiener Filter

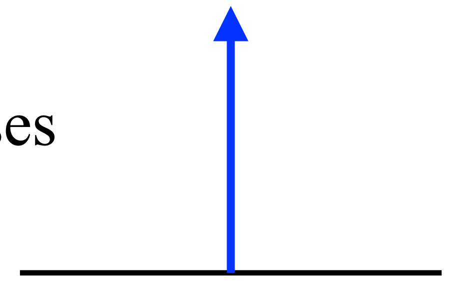
Outline

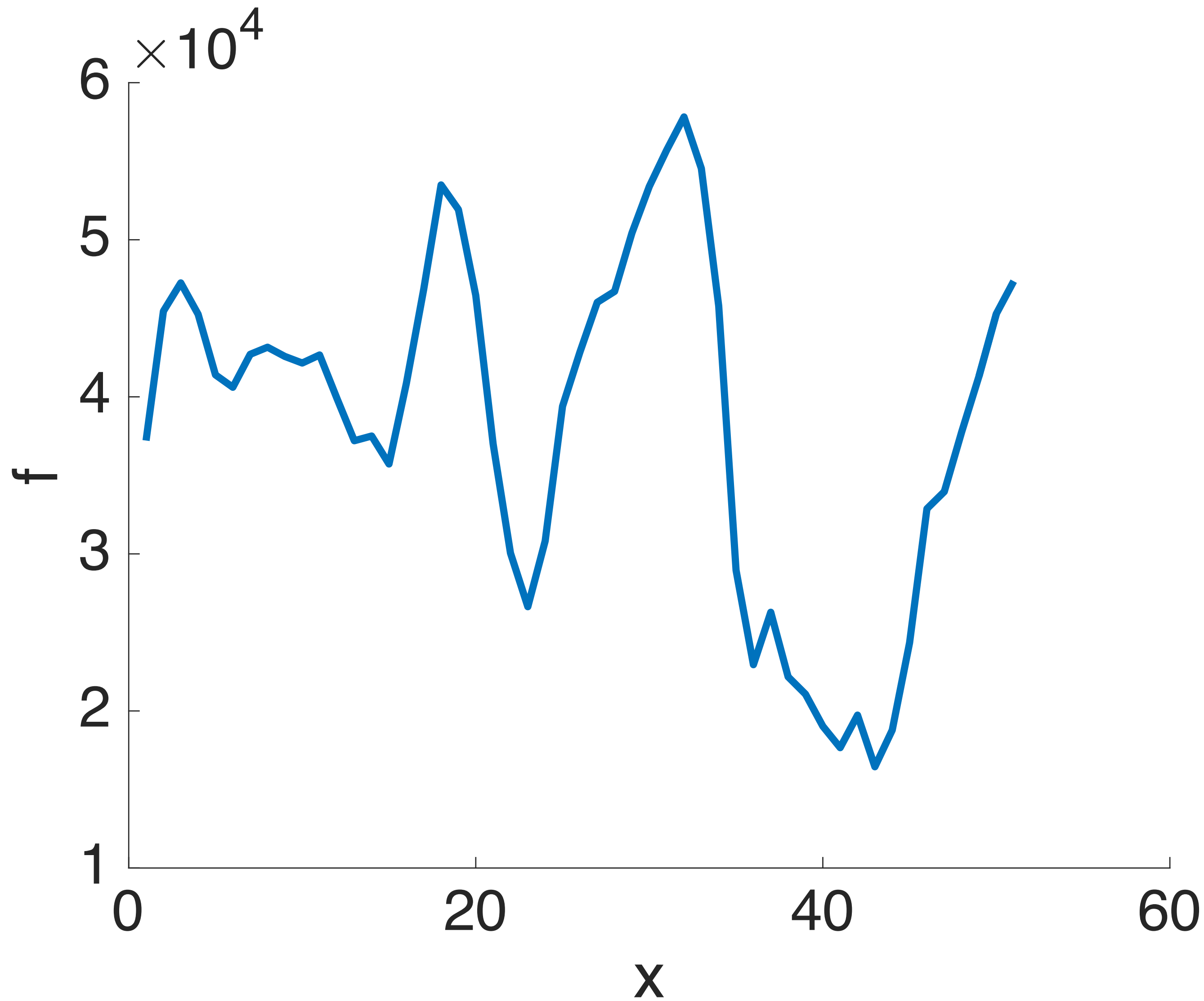
- **Linear Shift-Invariant Systems**
- The Fourier Transform
- The Wiener Filter

1D Signal Coding

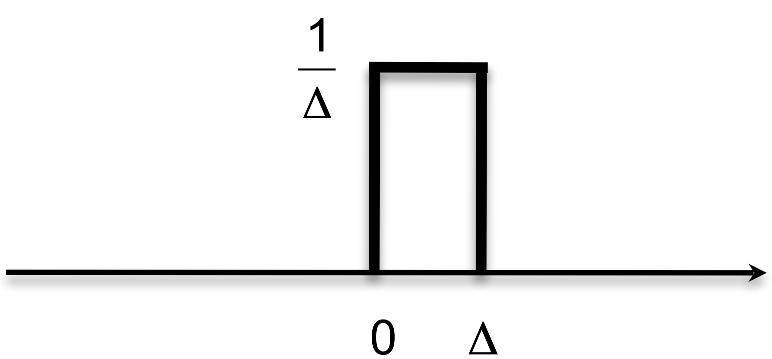
- A 1D signal (e.g., a slice of a luminance image $f(x)$ over horizontal location x) can be coded as a sequence of values

- This can also be viewed as a superposition of shifted and weighted impulses





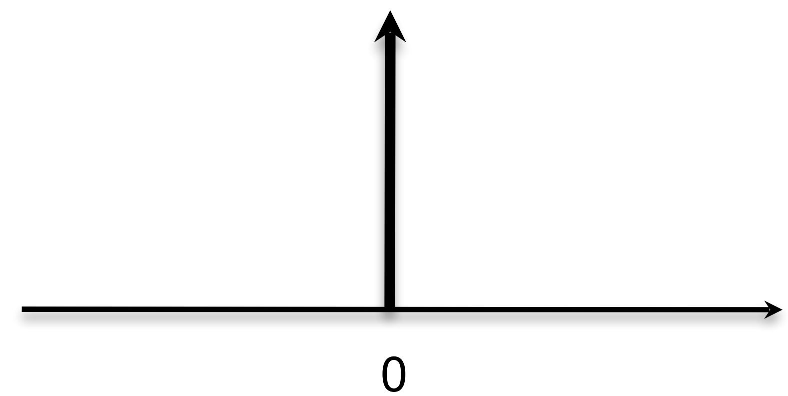
Impulse (Delta) Functions



Discrete pulse

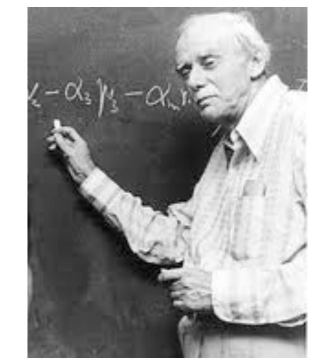
$$\delta_\Delta(x) = \begin{cases} \frac{1}{\Delta} \text{ if } 0 \le x \le \Delta \\ 0 \text{ otherwise} \end{cases}$$

$$\lim_{\Delta \to 0} \delta_\Delta(x)$$



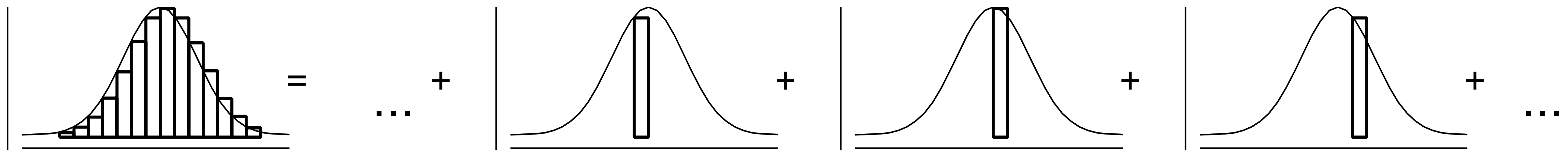
Dirac delta function

$$\delta(x) = \begin{cases} \infty \text{ if } x = 0 \\ 0 \text{ otherwise} \end{cases}$$



Paul Dirac
1902 - 1984

Representing a Signal with Impulses



$$f(x) \simeq \sum_{k=-\infty}^{\infty} f(k\Delta)\delta_\Delta(x - k\Delta)\Delta$$

$$f(x) = \lim_{\Delta \to 0} \sum_{k=-\infty}^{\infty} f(k\Delta)\delta_\Delta(x - k\Delta)\Delta$$

$$= \int_{-\infty}^{\infty} f(u)\delta(x-u)du$$

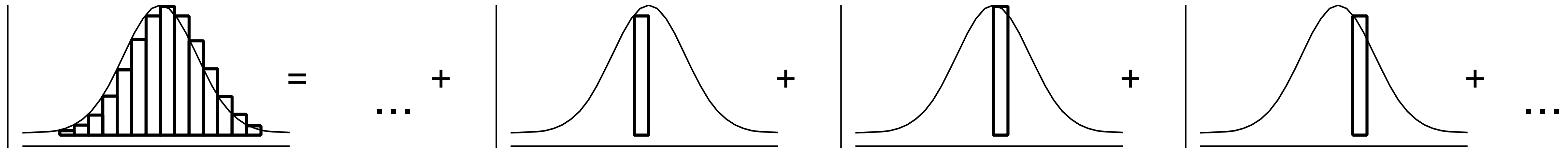
$$= f(x) * \delta(x)$$

Representing a Filter with Impulses

- Of course we can also code a filter $h(x)$ using impulses.

- This is why we refer to h(x) as the impulse response function of the filter

$$h(x) = h(x) * \delta(x)$$



Alternative Linear Codes

- The impulse code is not the only way to code a signal or a filter!
- In particular, there are many alternative linear codes, including
  - Fourier transforms
  - Discrete coding transforms (DCTs)
  - Wavelet transforms
- These linear codes are simply linear transformations of the impulse code.
- We begin with the Fourier code, which arises naturally from linear systems theory.

What is a linear system?

- A system h is linear if it satisfies the **principle of superposition**:

Additivity

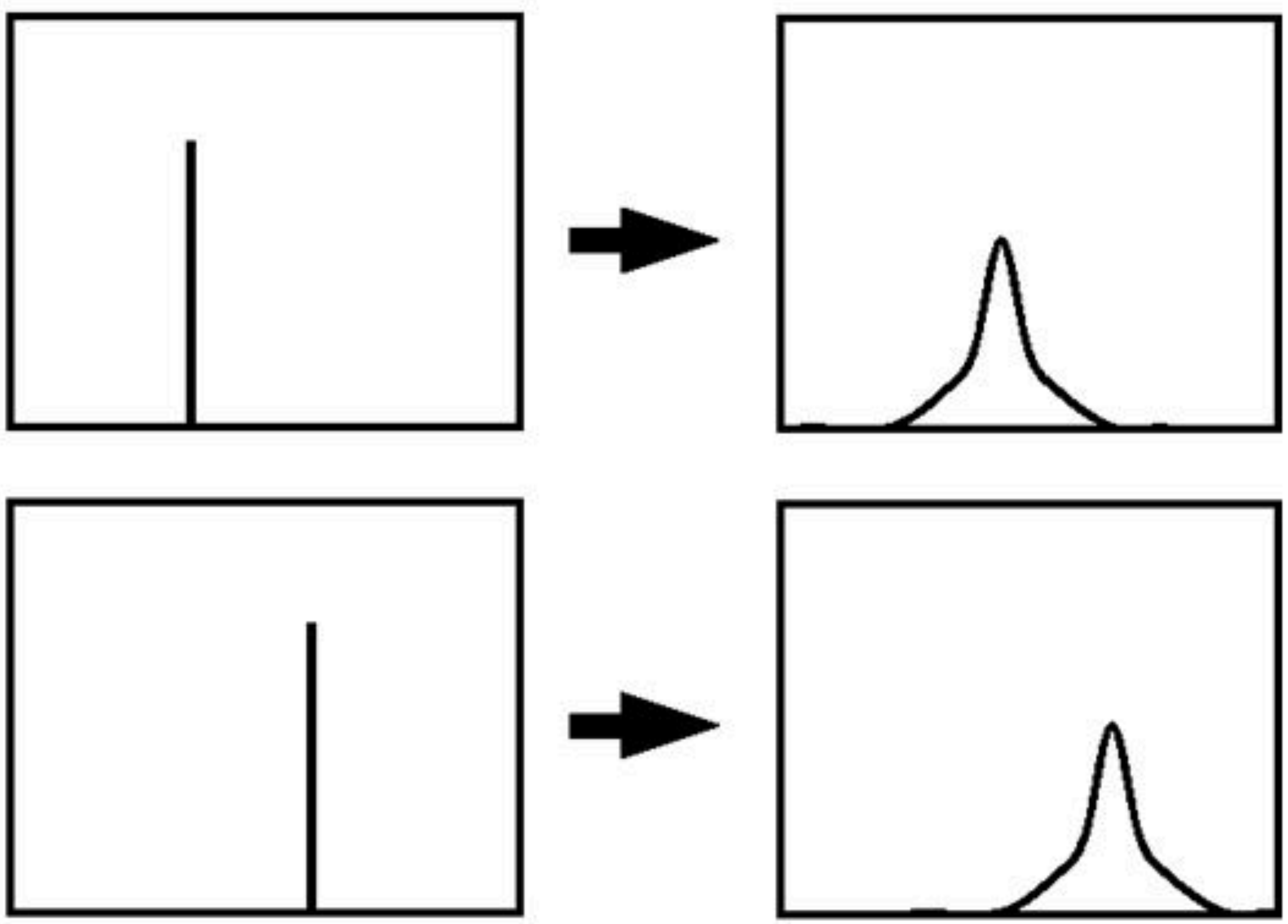
$$h(\alpha f_1 + \beta f_2) = \alpha h(f_1) + \beta h(f_2)$$

Homogeneity

Shift Invariance

- A system *h* is **shift-invariant** if a shift in the input produces an identical shift in the output:

$$g(x) = h\big(f(x)\big) \rightarrow g(x-u) = h\big(f(x-u)\big)$$



The Impulse Response Function

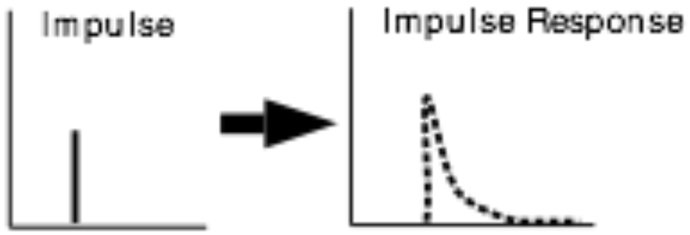
- The output of a linear shift-invariant system at x is a weighted sum of the input, where the weights are fixed relative to x.
- These filter weights are simply the reversed impulse response function.

$$h(x) = h(x) * \delta(x)$$

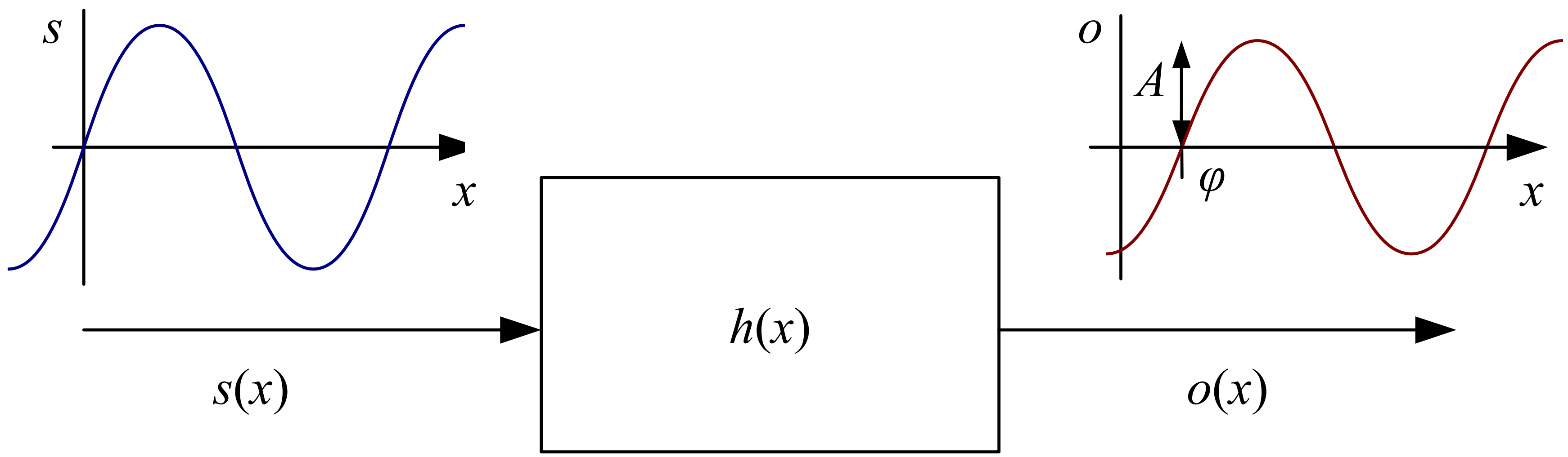
	past			present				future		
$\delta(s)$	0	0	0	1	0	0	0	0	0	input (impulse)
$h(x-s)$		1/8	1/4	1/2	→					weights
$h(x)$	0	0	0	1/2	1/4	1/8	0	0	0	output (impulse response)

Sinusoids

- When we input an impulse to a linear shift-invariant system, we get a complicated output (the impulse response)



- However, when we input a sinusoid, we get another sinusoid of the same frequency, but scaled and shifted in phase.



$s(x) = \sin(2\pi f x + \phi_i) = \sin(\omega x + \phi_i)$ $\qquad$ $o(x) = h(x) * s(x) = A \sin(\omega x + \phi_o)$

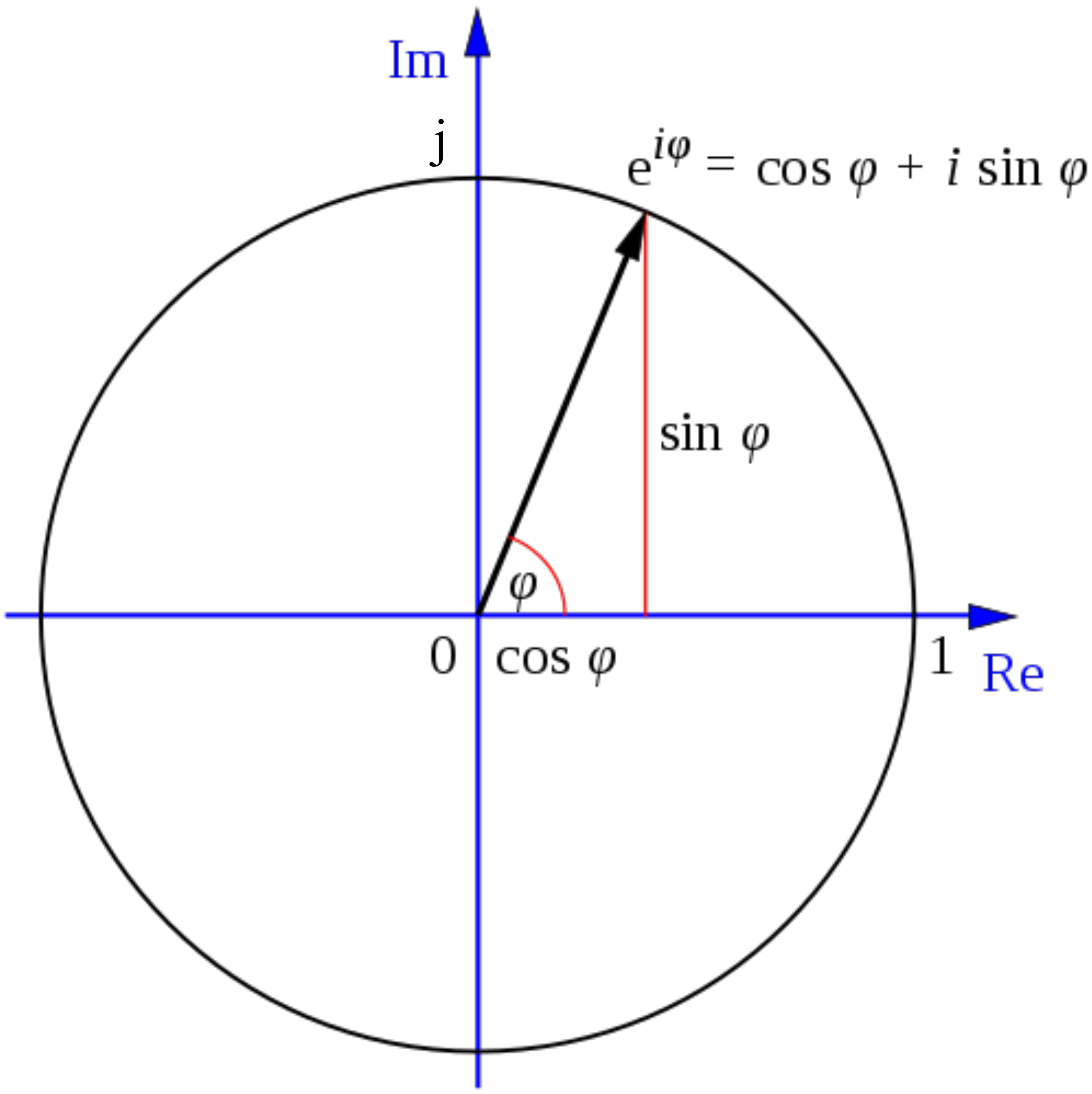
- This makes sinusoids a natural code for linear shift invariant systems.

Complex Sinusoids

- It is often convenient to work with complex sinusoids:

$$s(x) = e^{j\omega x} = \cos\omega x + j\sin\omega x$$

$$o(x) = h(x) * s(x) = Ae^{j\omega x + \phi}$$



Outline

- Linear Shift-Invariant Systems
- **The Fourier Transform**
- The Wiener Filter

Fourier Series

- We have already seen that any signal $f(x)$ or filter $h(x)$ can be expressed exactly as an infinite sum of impulses.

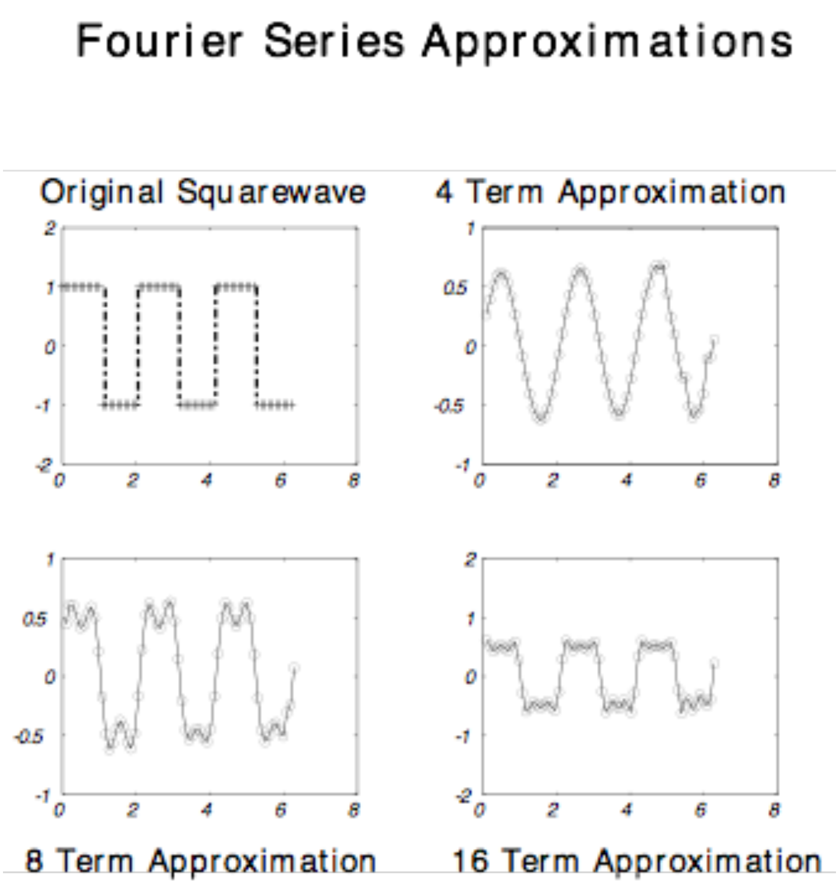
- It turns out that any signal can alternatively be expressed exactly as an infinite sum of sinusoids.

- This is known as a Fourier series.

- For a finite signal $f(x)$ defined on $[0, X]$, we have:

$$f(x) = \sum_{n=1}^{\infty} A_n \sin\left(2\pi n x / X + \phi_n\right)$$

Joseph Fourier
1768 - 1830



The Fourier Transform

- In the limit as X→∞, the Fourier series becomes the Fourier transform.

- The Fourier transform of a signal *f*(*x*) or filter *h*(*x*) is the response to a complex sinusoid at each frequency

$$H(\omega) = \mathcal{F}\{h(x)\} = Ae^{j\phi}$$

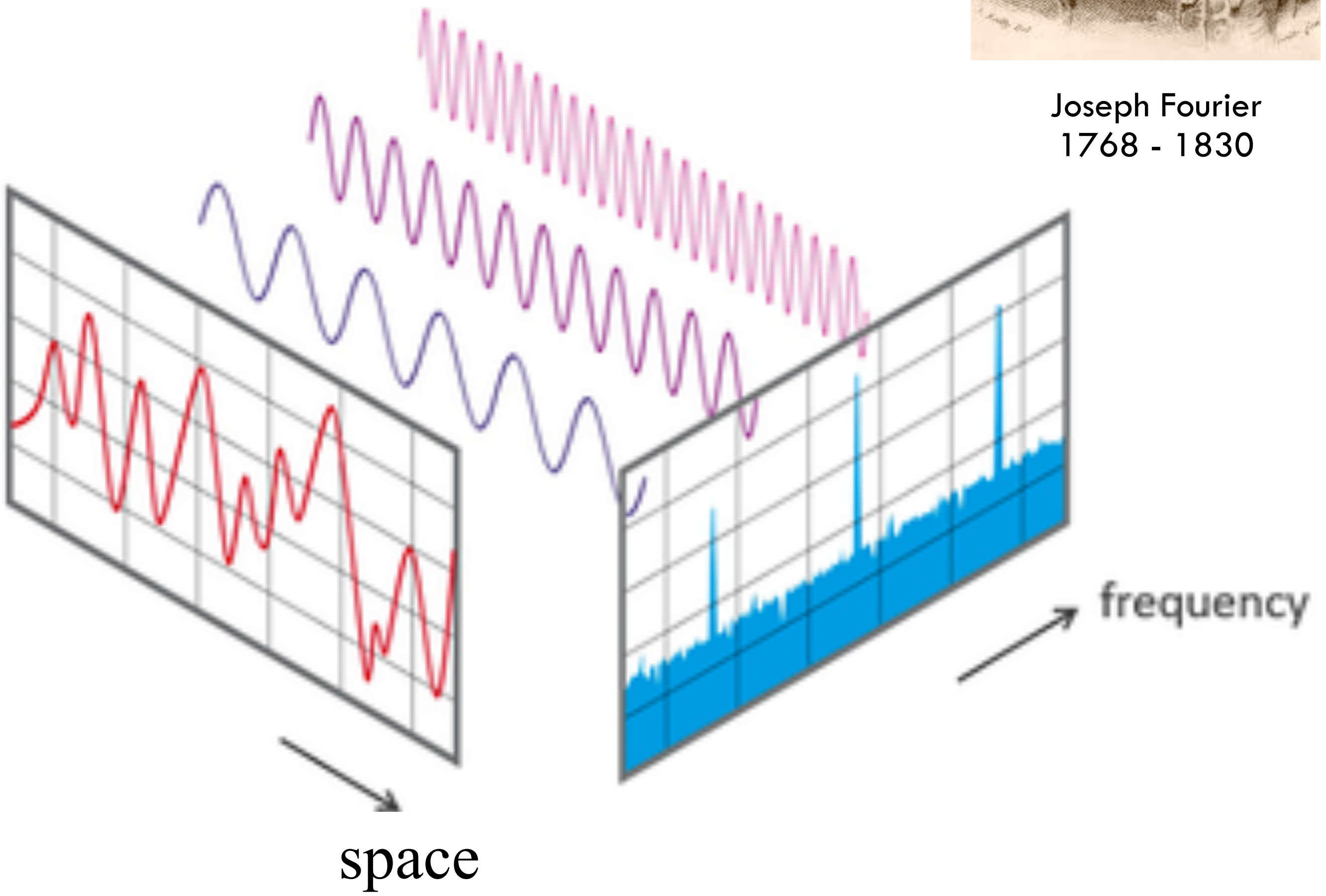
$$h(x) \overset{\mathcal{F}}{\leftrightarrow} H(\omega)$$

$H(\omega)$ is called the transfer function of the filter $h(x)$.

Joseph Fourier
1768 - 1830

- Continuous domain:

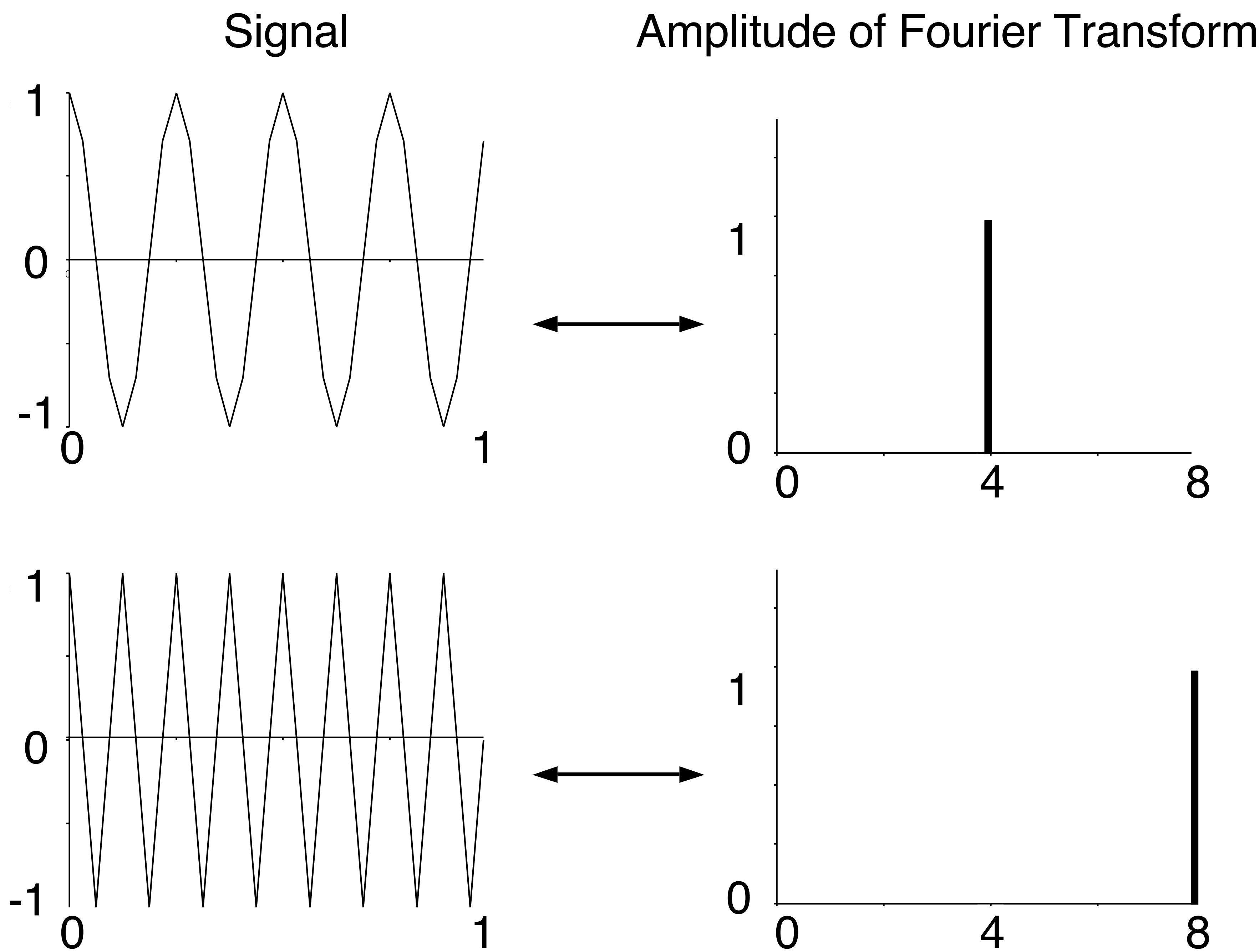
$$H(\omega) = \int_{-\infty}^{\infty} h(x) e^{-j\omega x} dx$$



Fourier Transforms

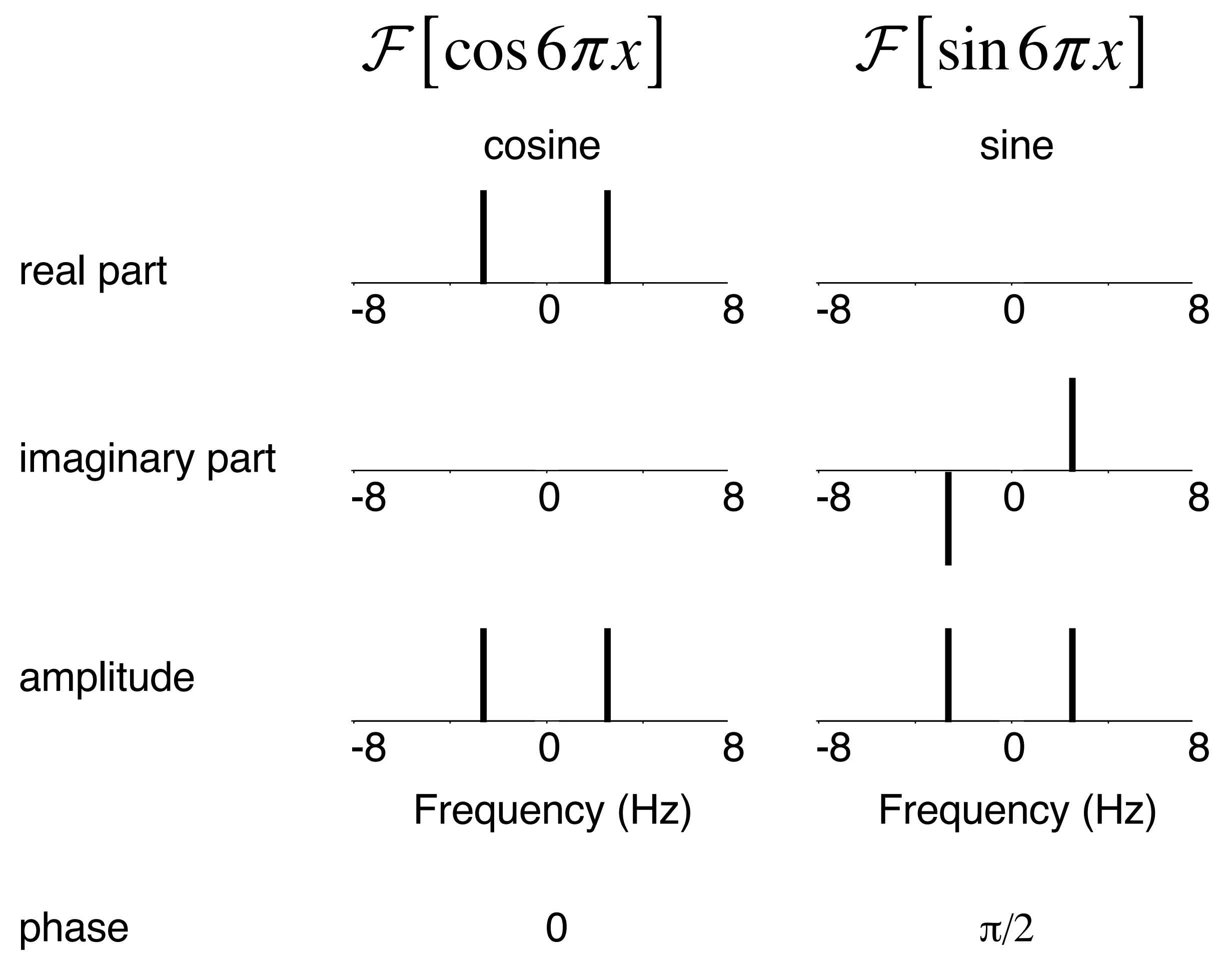
$\cos 8\pi x$

$\cos 16\pi x$



Amplitude & Phase

$$z = x + jy = Ae^{j\phi}, \text{ where } A = \sqrt{x^2 + y^2} \text{ and } \phi = \arctan\left(y/x\right)$$



End of Lecture
Oct 3, 2018

The Discrete Fourier Transform (DFT)

$$H(k) = \frac{1}{N}\sum_{x=0}^{N-1} h(x)e^{-j\frac{2\pi kx}{N}}$$

where N is the number of samples in the signal.

- Interpreting frequencies:
  - If N is odd:

$$k = 0 \Leftrightarrow \text{DC value (mean)}$$
$$k = 1 \Leftrightarrow 1 \text{ cycle per image, } 1/N \text{ cycles per pixel}$$
$$k = 2 \Leftrightarrow 2 \text{ cycles per image, } 2/N \text{ cycles per pixel}$$
$$\vdots$$
$$k = (N-1)/2 \Leftrightarrow (N-1)/2 \text{ cycles per image, } \frac{1}{2}\left(1-\frac{1}{N}\right) \text{ cycles per pixel (Nyquist limit)}$$
$$k = (N+1)/2 = N-(N-1)/2 \Leftrightarrow -(N-1)/2 \text{ cycles per image, } -\frac{1}{2}\left(1-\frac{1}{N}\right) \text{ cycles per pixel (Nyquist limit)}$$
$$\vdots$$
$$k = N-2 \Leftrightarrow -2 \text{ cycles per image, } -2/N \text{ cycles per pixel}$$
$$k = N-1 \Leftrightarrow -1 \text{ cycles per image, } -1/N \text{ cycles per pixel}$$

The Discrete Fourier Transform (DFT)

$$H(k) = \frac{1}{N}\sum_{x=0}^{N-1} h(x) e^{-j\frac{2\pi kx}{N}}$$

where N is the number of samples in the signal.

- Interpreting frequencies:
  - If N is even:

$k = 0 \Leftrightarrow$ DC value (mean)

$k = 1 \Leftrightarrow 1$ cycle per image, $1/N$ cycles per pixel

$k = 2 \Leftrightarrow 2$ cycles per image, $2/N$ cycles per pixel

$\vdots$

$k = N/2-1 \Leftrightarrow N/2-1$ cycles per image, $\frac{1}{2}-\frac{1}{N}$ cycles per pixel

$k = N/2 \Leftrightarrow N/2$ cycles per image, $\frac{1}{2}$ cycles per pixel (Nyquist limit)

$k = N/2+1 = N-(N/2-1) \Leftrightarrow -(N/2-1)$ cycles per image, $-\left(\frac{1}{2}-\frac{1}{N}\right)$ cycles per pixel

$\vdots$

$k = N-2 \Leftrightarrow -2$ cycles per image, $-2/N$ cycles per pixel

$k = N-1 \Leftrightarrow -1$ cycles per image, $-1/N$ cycles per pixel

The Discrete Fourier Transform (DFT)

$$H(k) = \frac{1}{N} \sum_{x=0}^{N-1} h(x) e^{-j \frac{2\pi k x}{N}}$$

where N is the number of samples in the signal.

- What is the computational complexity for computing the DFT?
  - Naïve: $O(N^2)$
  - Fast Fourier Transform (FFT): $O(N \log N)$

The Inverse Fourier Transform

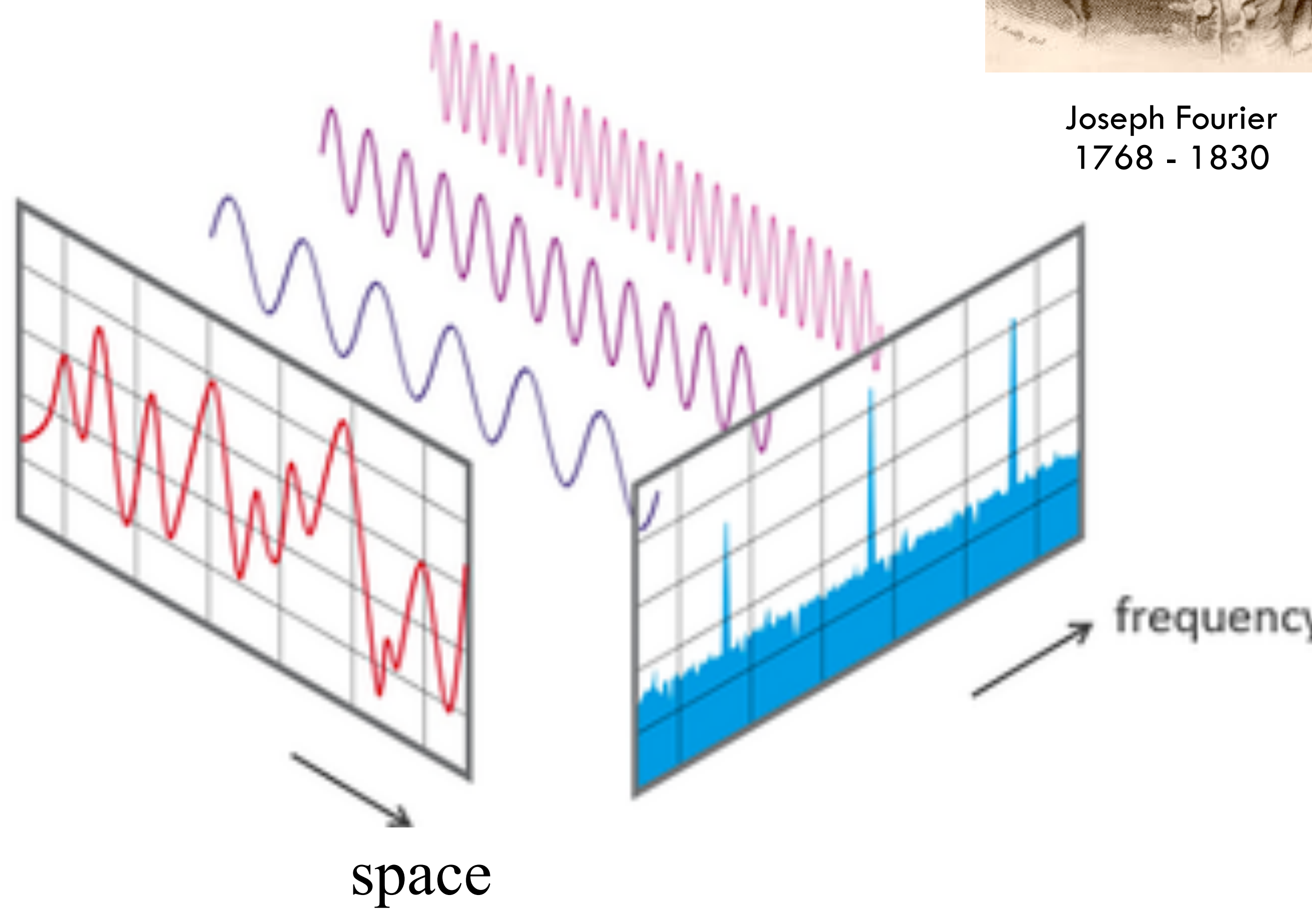
- Continuous domain:

$$h(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} H(\omega) e^{j\omega x} \, d\omega$$

- Discrete domain:

$$h(x) = \frac{1}{N} \sum_{k=-N/2}^{N/2} H(k) e^{j\frac{2\pi k x}{N}}$$

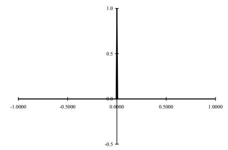
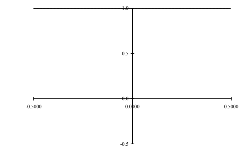
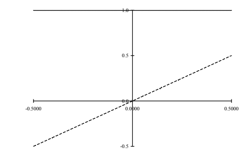
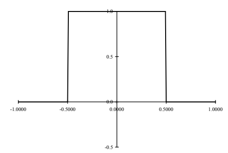
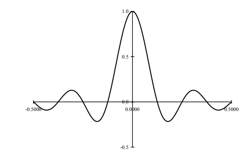
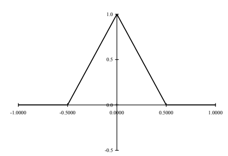
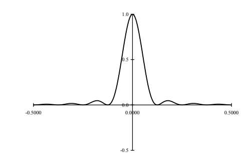
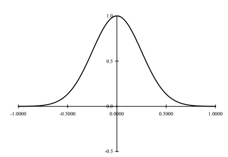
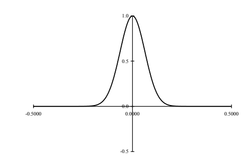
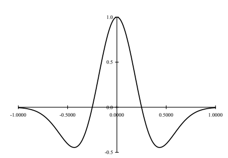
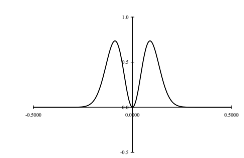
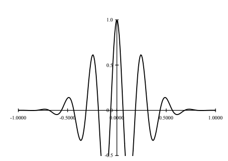
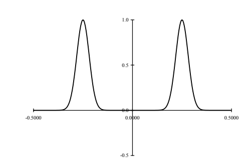
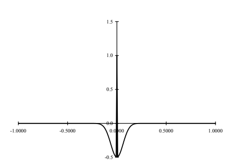
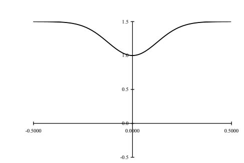
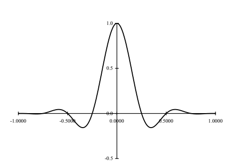
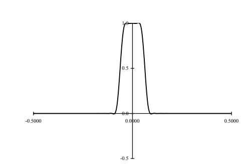
Joseph Fourier
1768 - 1830



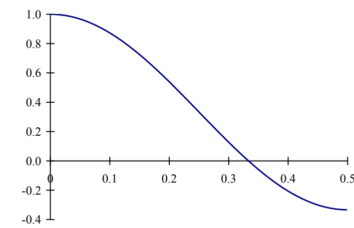
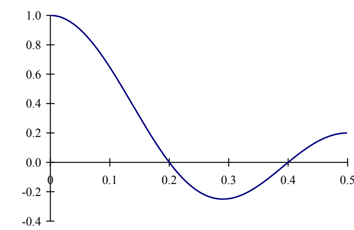
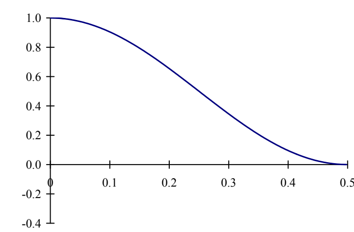
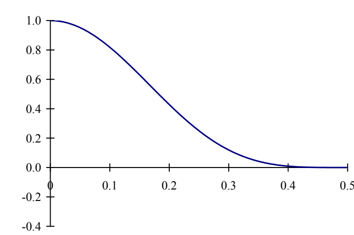
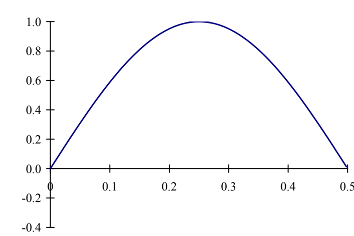
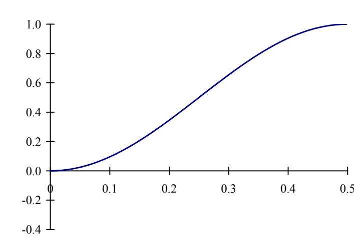
Properties of the Fourier Transform

Property	Signal		Transform
superposition	$f_1(x) + f_2(x)$		$F_1(\omega) + F_2(\omega)$
shift	$f(x - x_0)$		$F(\omega)e^{-j\omega x_0}$
reversal	$f(-x)$		$F^*(\omega)$
convolution	$f(x) * h(x)$		$F(\omega)H(\omega)$
correlation	$f(x) \otimes h(x)$		$F(\omega)H^*(\omega)$
multiplication	$f(x)h(x)$		$F(\omega) * H(\omega)$
differentiation	$f'(x)$		$j\omega F(\omega)$
domain scaling	$f(ax)$		$1/aF(\omega/a)$
real images	$f(x) = f^*(x)$	$\Leftrightarrow$	$F(\omega) = F(-\omega)$
Parseval's Theorem	$\sum_x [f(x)]^2$	$=$	$\sum_\omega [F(\omega)]^2$

Fourier Pairs

Name	Signal			Transform		
impulse		$\delta(x)$	$\Leftrightarrow$	1		
shifted impulse		$\delta(x-u)$	$\Leftrightarrow$	$e^{-j\omega u}$		(phase plot)
box filter		$\text{box}(x/a)$	$\Leftrightarrow$	$a\text{sinc}(a\omega)$		
tent		$\text{tent}(x/a)$	$\Leftrightarrow$	$a\text{sinc}^2(a\omega)$		
Gaussian		$G(x;\sigma)$	$\Leftrightarrow$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega;\sigma^{-1})$		
Laplacian of Gaussian		$(\frac{x^2}{\sigma^4}-\frac{1}{\sigma^2})G(x;\sigma)$	$\Leftrightarrow$	$-\frac{\sqrt{2\pi}}{\sigma}\omega^2 G(\omega;\sigma^{-1})$		
Gabor		$\cos(\omega_0 x)G(x;\sigma)$	$\Leftrightarrow$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega\pm\omega_0;\sigma^{-1})$		
unsharp mask		$(1+\gamma)\delta(x)$ $-\gamma G(x;\sigma)$	$\Leftrightarrow$	$(1+\gamma)-$ $\frac{\sqrt{2\pi}\gamma}{\sigma}G(\omega;\sigma^{-1})$		
windowed sinc		$\text{rcos}(x/(aW))$ $\text{sinc}(x/a)$	$\Leftrightarrow$	(see Figure 3.29)		

Fourier transforms of simple filters

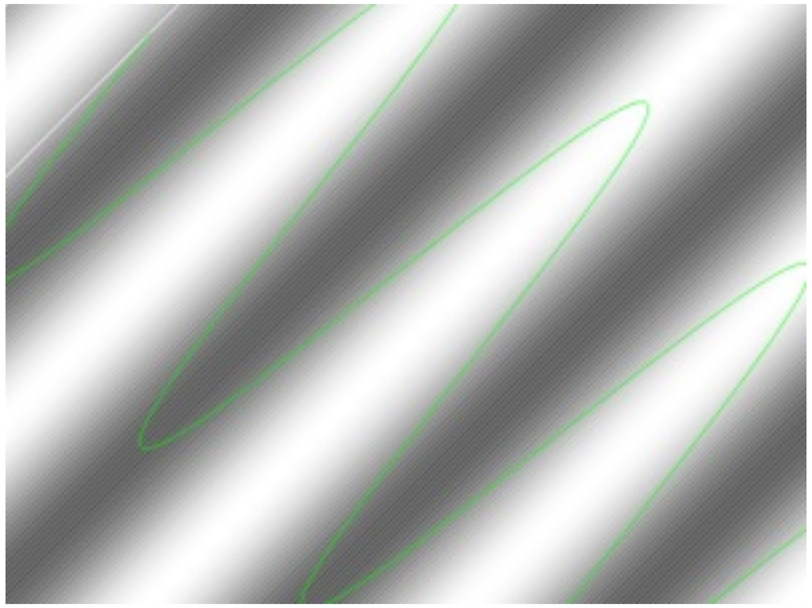
Name	Kernel	Transform	Plot
box-3	$\frac{1}{3}\,[\,1\;\;1\;\;1\,]$	$\frac{1}{3}(1+2\cos\omega)$	
box-5	$\frac{1}{5}\,[\,1\;\;1\;\;1\;\;1\;\;1\,]$	$\frac{1}{5}(1+2\cos\omega+2\cos 2\omega)$	
linear	$\frac{1}{4}\,[\,1\;\;2\;\;1\,]$	$\frac{1}{2}(1+\cos\omega)$	
binomial	$\frac{1}{16}\,[\,1\;\;4\;\;6\;\;4\;\;1\,]$	$\frac{1}{4}(1+\cos\omega)^2$	
Sobel	$\frac{1}{2}\,[\,-1\;\;0\;\;1\,]$	$\sin\omega$	
corner	$\frac{1}{2}\,[\,-1\;\;2\;\;-1\,]$	$\frac{1}{2}(1-\cos\omega)$	

The 2D Fourier Transform

- The extension to 2D images and filters is straightforward.

- The 2D Fourier transform tabulates the amplitude and phase of sinusoidal gratings for all combinations of horizontal and vertical frequency:

$$s(x,y) = \sin(\omega_x x + \omega_y y)$$



$$H(\omega_x, \omega_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x,y) e^{-j(\omega_x x + \omega_y y)} dx\, dy.$$

$$H(k_x, k_y) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} h(x,y) e^{-j2\pi\frac{k_x x + k_y y}{MN}}$$

Outline

- Linear Shift-Invariant Systems
- The Fourier Transform
- **The Wiener Filter**

Noise

- Images formed by a camera or the eye are corrupted by noise.
- This noise can often be approximated as a zero-mean, additive and stationary random process.

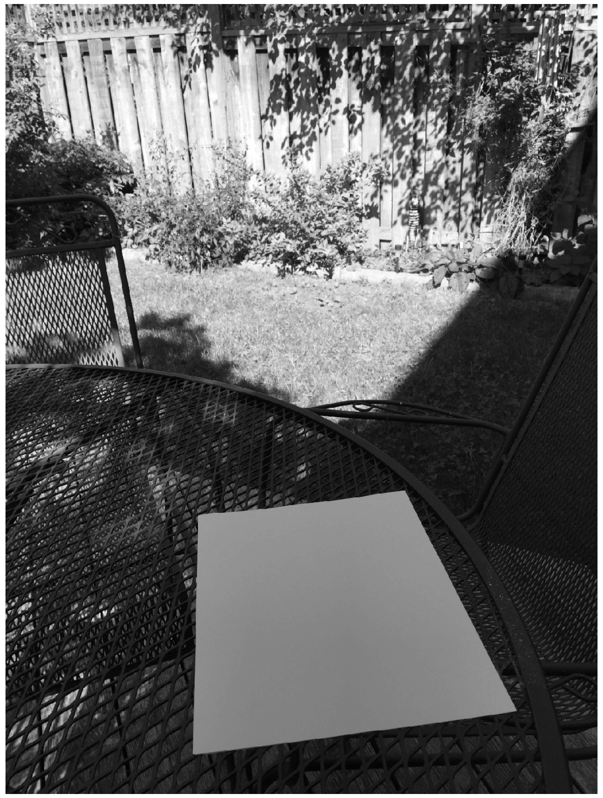
$$f(x,y) = g(x,y) + n(x,y)$$

Optical image

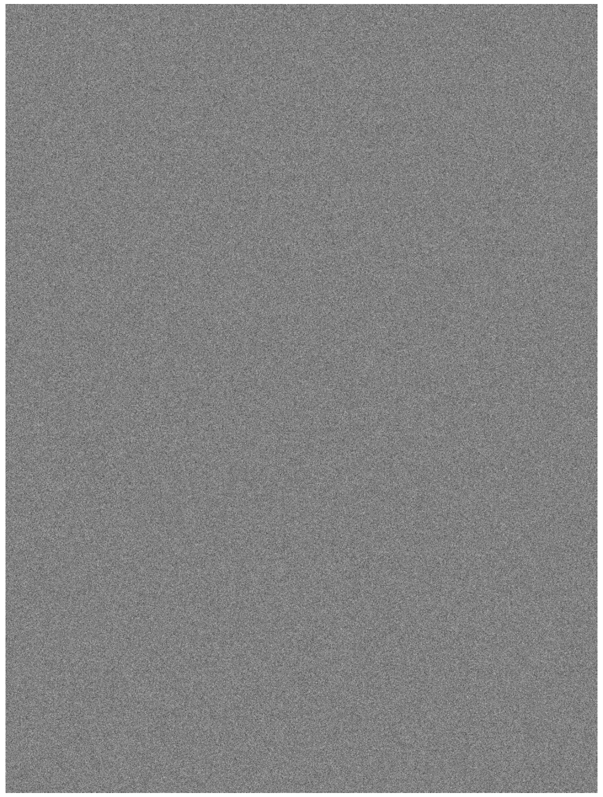
Noisy sensed image

Added noise

=



+



Noise Filtering

$$f(x,y) = g(x,y) + n(x,y)$$

- Denoising is a core problem in image processing.
- The linear systems solution to this problem is well understood.
- The problem is to find the optimal filter $h(x,y)$ that will maximize the accuracy of the filtered image in the least squares sense.
- By the convolution theorem, this is equivalent to identifying the optimal transfer function $H(\omega_x, \omega_y)$

$$h(x,y) * f(x,y) \Leftrightarrow H(\omega_x, \omega_y) F(\omega_x, \omega_y)$$

Probabilistic Model

$$f(x,y) = g(x,y) + n(x,y)$$

- To solve this problem, we assume that the optical image g(x,y) and the noise n(x,y) are both independent, stationary, random processes whose *power spectral densities* are known
  - Power spectral densities:

$$P_f(\omega_x,\omega_y) = \left\langle \left| F(\omega_x,\omega_y) \right|^2 \right\rangle = \mathbb{E}\left[\left| F(\omega_x,\omega_y) \right|^2 \right]$$

$$P_g(\omega_x,\omega_y) = \left\langle \left| G(\omega_x,\omega_y) \right|^2 \right\rangle = \mathbb{E}\left[\left| G(\omega_x,\omega_y) \right|^2 \right]$$

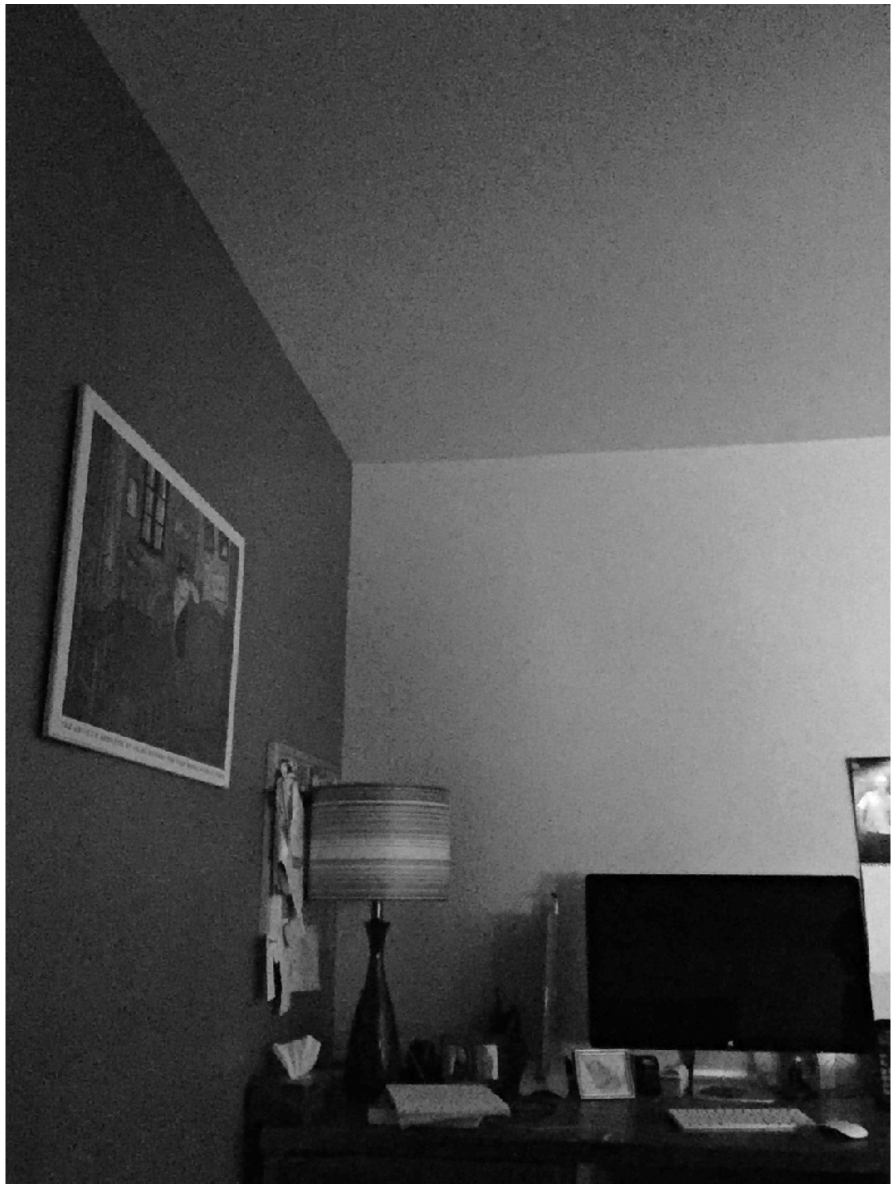
$$P_n(\omega_x,\omega_y) = \left\langle \left| N(\omega_x,\omega_y) \right|^2 \right\rangle = \mathbb{E}\left[\left| N(\omega_x,\omega_y) \right|^2 \right]$$

End of Lecture
Oct 15, 2018

Power Spectral Density

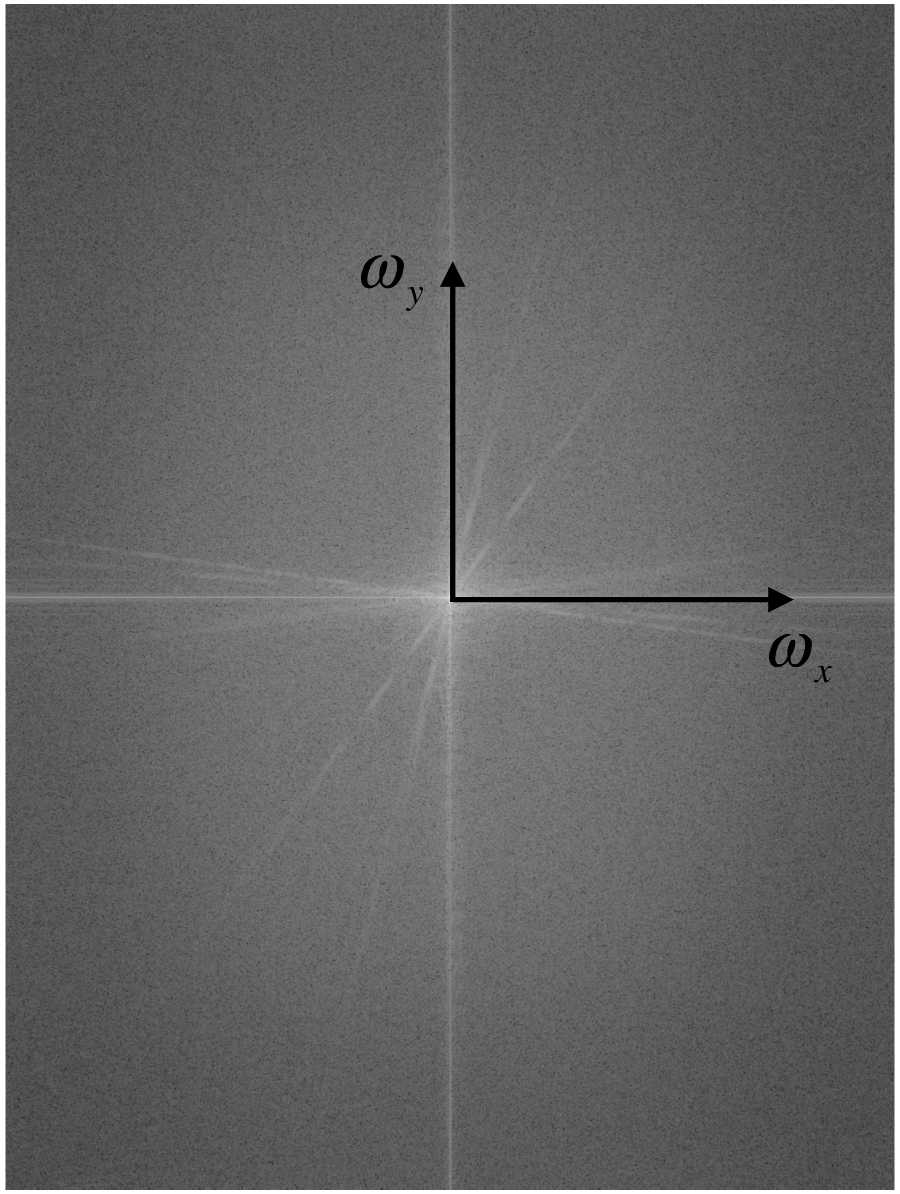
- Natural images tend to be lowpass - most of the energy is in the low spatial frequencies.

Image



$g(x,y)$

Log Fourier Energy

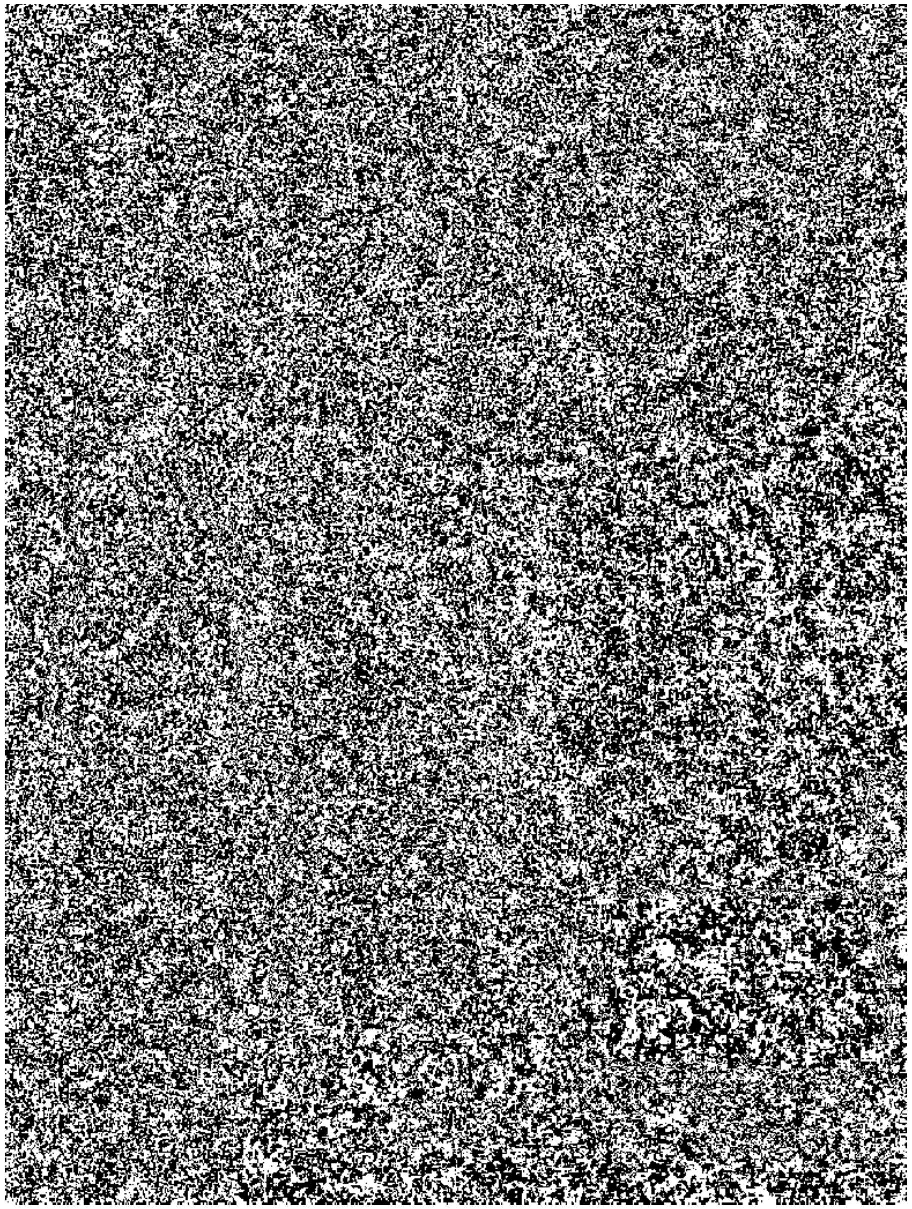


$\log\left|G(\omega_x,\omega_y)\right|^2$

Noise Spectral Density

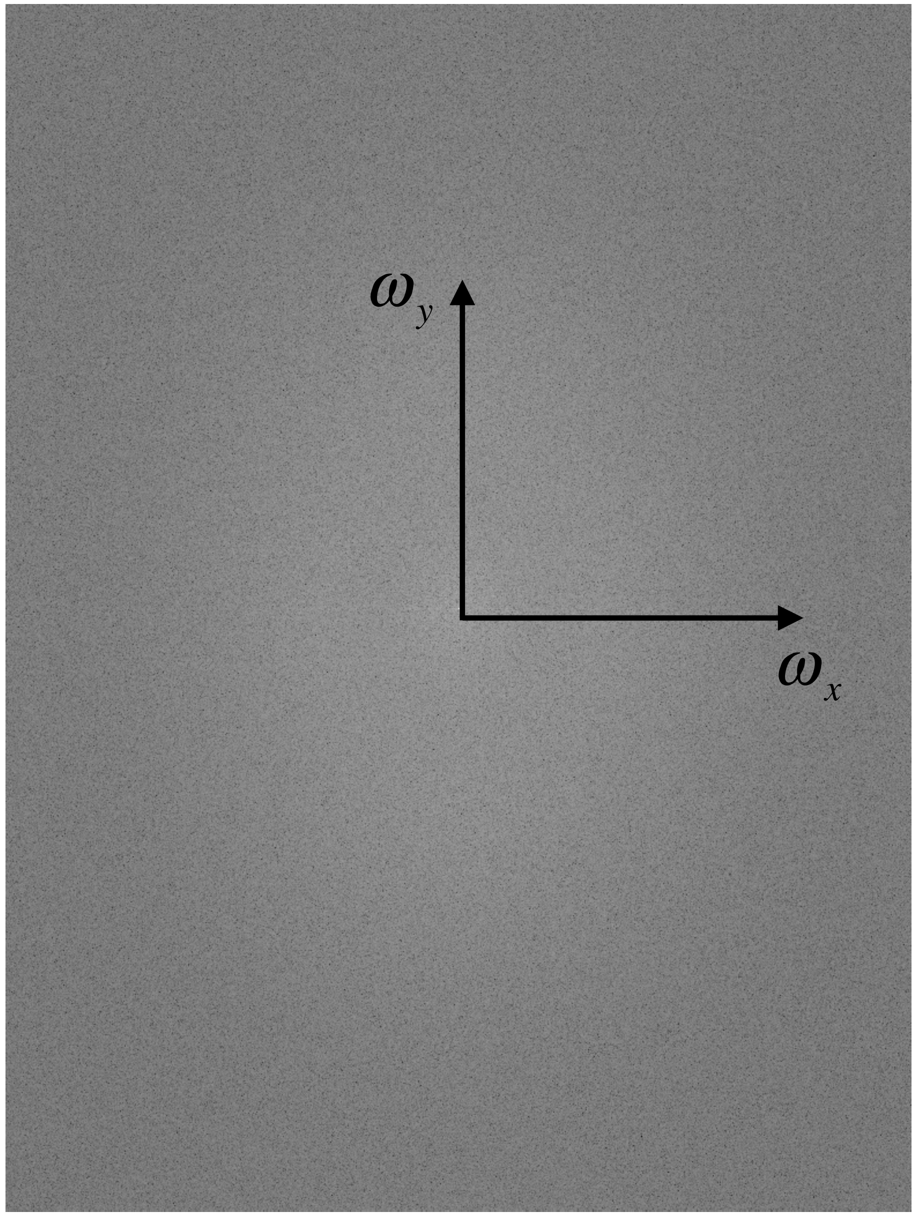
- In contrast, the expected energy in image noise tends to be more flat (white) across spatial frequency

Noise



$n(x,y)$

Log Fourier Energy



$\log\left|N(\omega_x,\omega_y)\right|^2$

The Wiener Filter

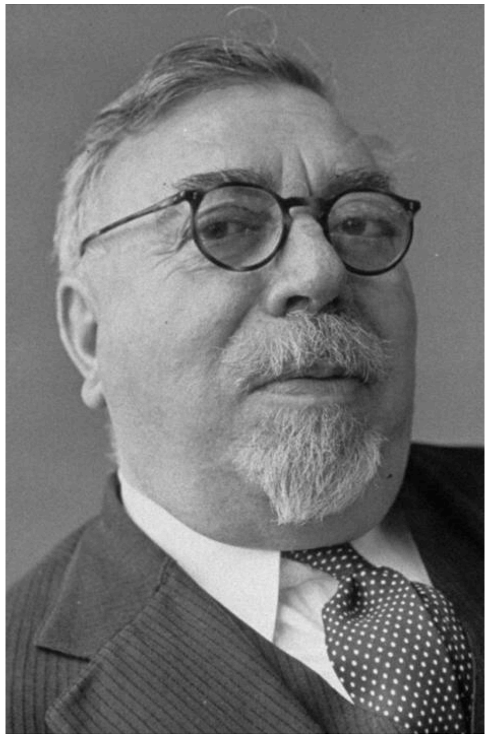
- When the frequency distribution of the image energy and the noise energy differ, we can improve the signal-to-noise ratio (SNR) by boosting the Fourier amplitudes where the image is strong relative to the noise and attenuating the Fourier amplitudes where it is relatively weak.
- Typically this means a lowpass filter.
- The Wiener filter is given by

$$H(\omega_x,\omega_y) = \frac{P_g(\omega_x,\omega_y)}{P_f(\omega_x,\omega_y)} = \frac{P_g(\omega_x,\omega_y)}{P_g(\omega_x,\omega_y)+P_n(\omega_x,\omega_y)}, \text{ where}$$

$P_f(\omega_x,\omega_y)$ is the power spectral density of the noisy sensed image

$P_g(\omega_x,\omega_y)$ is the power spectral density of the optical image before noise was added

$P_n(\omega_x,\omega_y)$ is the power spectral density of the noise



Norbert Wiener 1894 - 1964

The Wiener Filter

$$H(\omega_x,\omega_y) = \frac{P_g(\omega_x,\omega_y)}{P_f(\omega_x,\omega_y)} = \frac{P_g(\omega_x,\omega_y)}{P_g(\omega_x,\omega_y)+P_n(\omega_x,\omega_y)}, \text{ where}$$

$P_f(\omega_x,\omega_y)$ is the power spectral density of the noisy sensed image

$P_g(\omega_x,\omega_y)$ is the power spectral density of the optical image before noise was added

$P_n(\omega_x,\omega_y)$ is the power spectral density of the noise

- The Wiener filter minimizes the expected mean square error (MSE) of the estimated image relative to the original image before noise was added.
- It is the optimal linear shift-invariant solution to this problem
- Note that this optimality is general - it does not depend upon either the noise or the image being Gaussian. (Be careful with the textbook here.)

Estimating the Wiener Filter

$$H(\omega_x,\omega_y) = \frac{P_g(\omega_x,\omega_y)}{P_f(\omega_x,\omega_y)} = \frac{P_g(\omega_x,\omega_y)}{P_g(\omega_x,\omega_y)+P_n(\omega_x,\omega_y)}, \text{ where}$$

$P_f(\omega_x,\omega_y)$ is the power spectral density of the noisy sensed image

$P_g(\omega_x,\omega_y)$ is the power spectral density of the optical image before noise was added

$P_n(\omega_x,\omega_y)$ is the power spectral density of the noise

- To calculate the Wiener filter we need to know the power spectral density of the optical image and of the noise.
- Typically, we employ simple approximations.

Wiener Filter Example

$$H(\omega_x,\omega_y) = \frac{P_g(\omega_x,\omega_y)}{P_f(\omega_x,\omega_y)} = \frac{P_g(\omega_x,\omega_y)}{P_g(\omega_x,\omega_y)+P_n(\omega_x,\omega_y)}, \text{ where}$$

$P_f(\omega_x,\omega_y)$ is the power spectral density of the noisy sensed image

$P_g(\omega_x,\omega_y)$ is the power spectral density of the optical image before noise was added

$P_n(\omega_x,\omega_y)$ is the power spectral density of the noise

- Assume isotropic spectral densities for both image and noise
  - Image spectral density is lowpass

    $$P_g(\omega_x,\omega_y) = \frac{\alpha^2}{\omega^2}, \text{ where } \omega^2 = \omega_x^2 + \omega_y^2$$

  - Noise spectral density is white

    $$P_n(\omega_x,\omega_y) = \sigma_n^2$$

  - Then

    $$H(\omega_x,\omega_y) = \frac{(\alpha/\omega)^2}{(\alpha/\omega)^2+\sigma_n^2} = \frac{1}{1+(\omega/\beta)^2}, \text{ where } \beta = \alpha/\sigma_n \text{ is the SNR.}$$

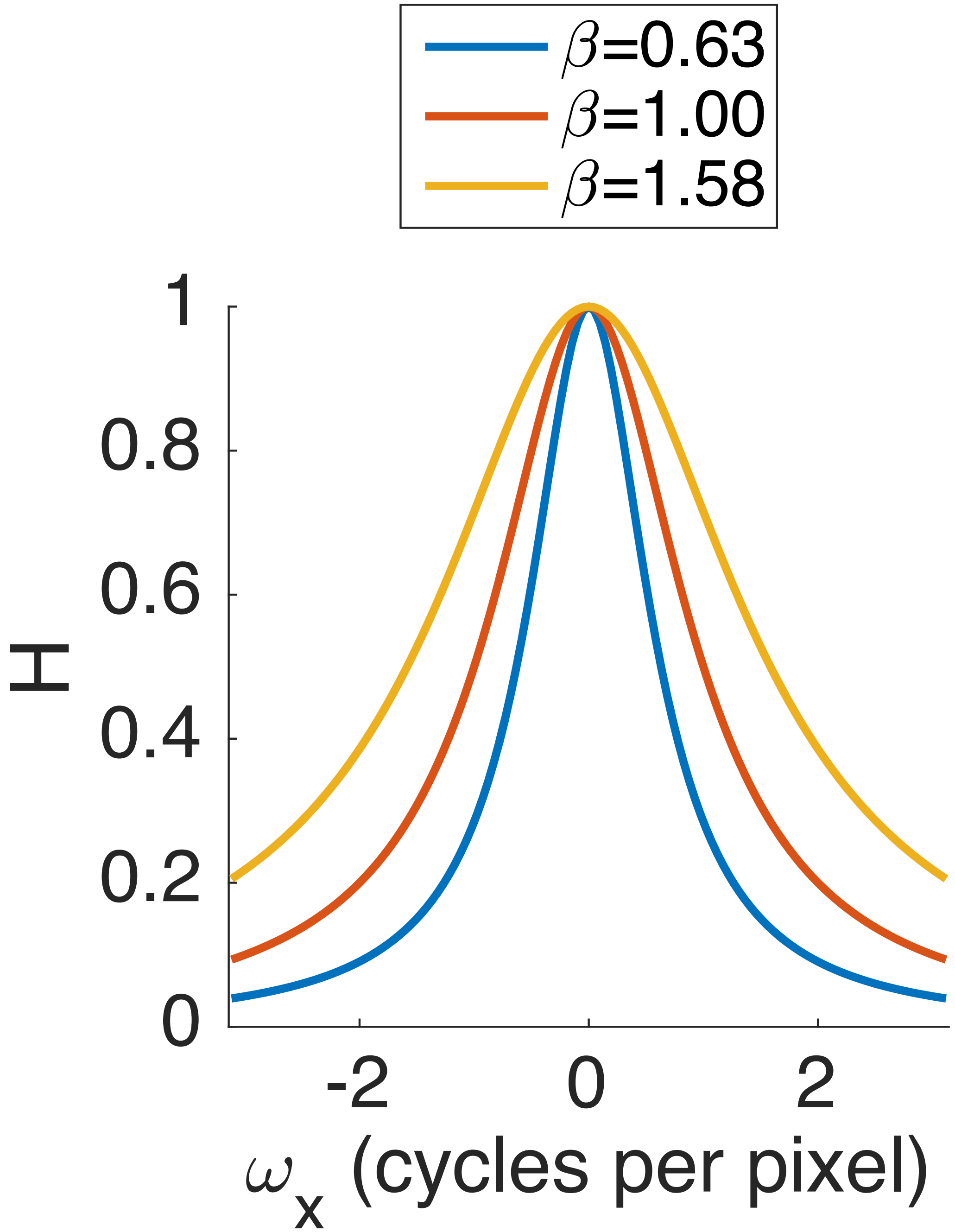
Field, 1987

Wiener Filter Example

$$H(\omega_x,\omega_y)=\frac{1}{1+(\omega/\beta)^2},\ \text{where } \beta=\alpha/\sigma_n \text{ is the SNR.}$$

- Observe that:

$$\lim_{\beta\to\infty} H(\omega_x,\omega_y)=1 \qquad \lim_{\beta\to 0} H(\omega_x,\omega_y)=\left(\frac{\beta}{\omega}\right)^2$$



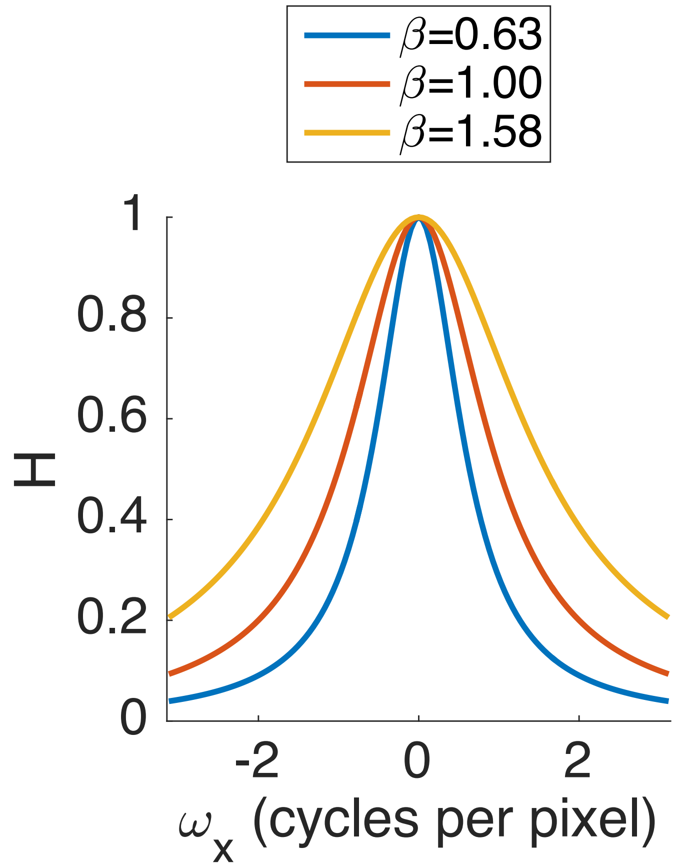
Wiener Filter Example

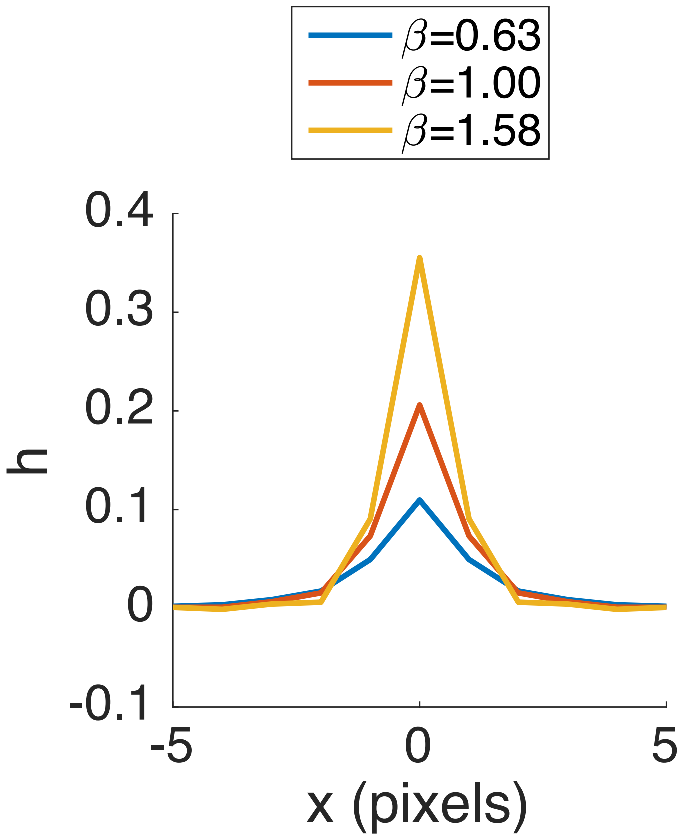
$$H\left(\omega_x,\omega_y\right)=\frac{1}{1+(\omega/\beta)^2},$$ where $\beta=\alpha/\sigma_n$ is the SNR.

❖ Note that:

- h(r) is the inverse Hankel transform of H(ω), not the Fourier transform.
- h(r) has no analytic form, but the discrete form of h(x, y) can be determined by taking the inverse Fourier transform of $H(\omega_x,\omega_y)$.

  $\left(\text{The Hankel transform of } \frac{\beta^2}{2\pi}e^{-\beta r} \text{ is actually } \frac{1}{\left(1+(\omega/\beta)^2\right)^{3/2}}\right)$





State of the Art

- Deep convolutional neural networks
  - Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L. (2017). Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. IEEE Transactions on Image Processing, 26(7):3142–3155.
  - Wang, R. and Tao, D. (2016). Non-local auto-encoder with collaborative stabilization for image restoration. IEEE Transactions on Image Processing, 25(5):2117–2129.
- Nonlinear filtering with learned parameters
  - Chen, Y. and Pock, T. (2017). Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6):1256–1272.

Outline

- Linear Shift-Invariant Systems
- The Fourier Transform
- The Wiener Filter