Median Operator

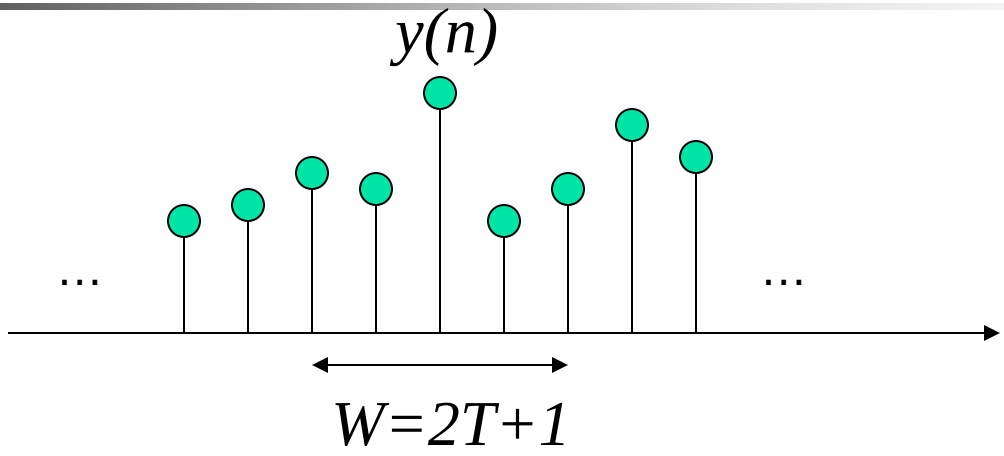
- Given a sequence of numbers $\{y_1,\ldots,y_N\}$
  - Mean: average of N numbers
  - Min: minimum of N numbers
  - Max: maximum of N numbers
  - Median: half-way of N numbers

Example $\vec{y} = [50,0,52,255,54,55,56]$

$$\text{sorted } \vec{y} = [0,50,52,54,55,56,255]$$

$$median(\vec{y}) = 54$$

1D Median Filtering



$$\hat{x}(n) = median[y(n-T),..., y(n),..., y(n+T)]$$

MATLAB command: x=median(y(n-T:n+T));

Note: median operator is nonlinear

Numerical Example

T=1:

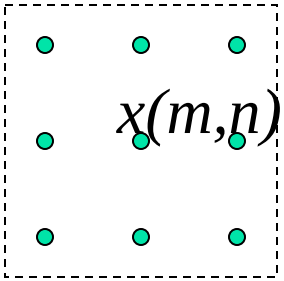
$$\vec{y} = [50,0,52,255,54,55,56]$$

Boundary Padding

$$\vec{y} = [\mathbf{50},50,0,52,255,54,55,56,\mathbf{56}]$$

$$\hat{\vec{x}} = [50,50,52,54,54,55,55]$$

2D Median Filtering



W: (2T+1)-by-(2T+1) window

$$\hat{x}(m,n) = median[y(m-T,n-T),...,y(m-T,n+T),...,$$
$$y(m,n),...,y(m+T,n-T),...,y(m+T,n+T)]$$

MATLAB command: x=medfilt2(y,[2*T+1,2*T+1]);

Numerical Example

225	225	225	226	226	226	226	226
225	225	255	226	226	226	225	226
226	226	225	226	**0**	226	226	255
255	226	225	**0**	226	226	226	226
225	255	**0**	225	226	226	226	255
255	225	224	226	226	0	225	226
226	225	225	226	255	226	226	228
226	226	225	226	226	226	226	226

Y

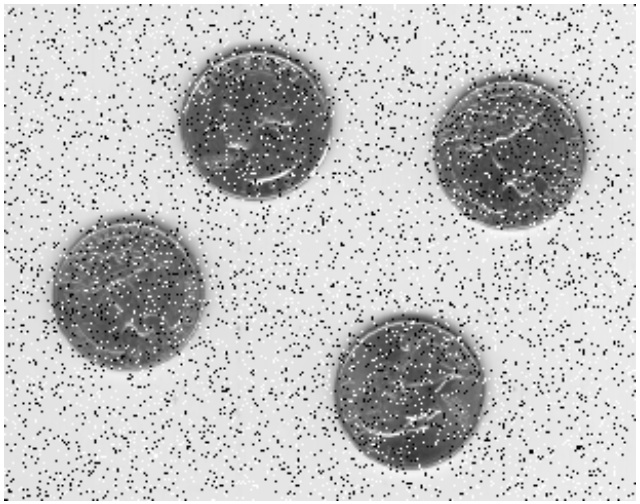
0	225	225	226	226	226	226	226
225	225	226	226	226	226	226	226
225	226	226	226	226	226	226	226
226	226	225	**225**	226	226	226	226
225	225	225	225	226	226	226	226
225	225	225	226	226	226	226	226
225	225	225	226	226	226	226	226
226	226	226	226	226	226	226	226

$\hat{X}$

Sorted: [0, 0, 0, 225, **225**, 225, 226, 226, 226]

Image Example

P=0.1



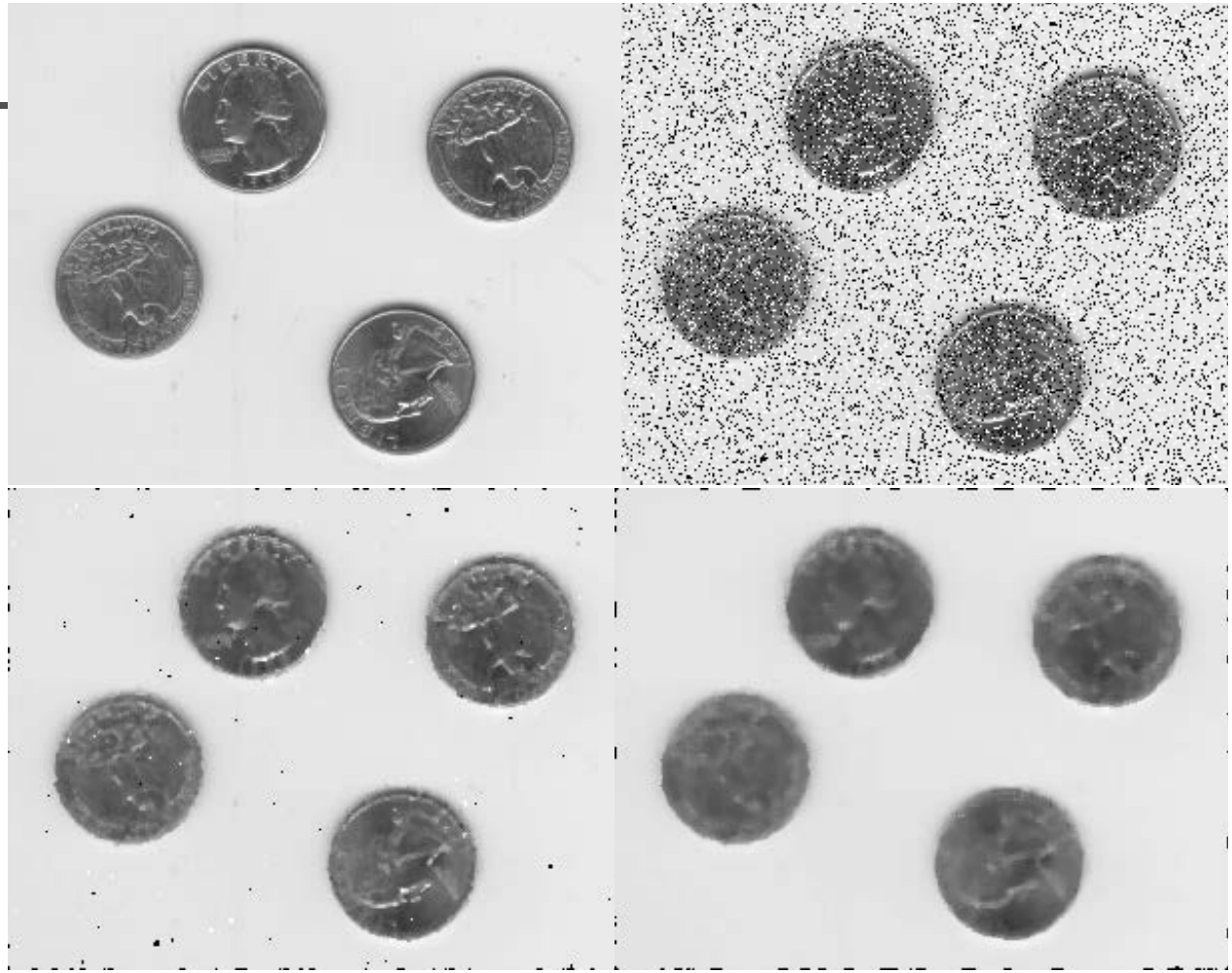
Noisy image Y

denoised image $\hat{X}$

3-by-3 window

Image Example (Con't)

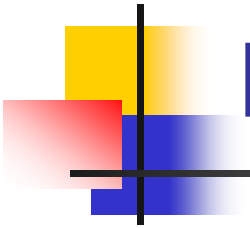
clean



noisy
(p=0.2)

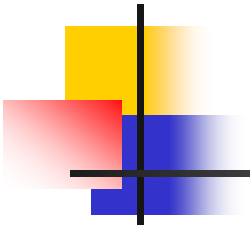
3-by-3 window

5-by-5 window



Reflections

- What is good about median operation?
  - Since we know impulse noise appears as black (minimum) or white (maximum) dots, taking median effectively suppresses the noise
- What is bad about median operation?
  - It affects clean pixels as well
  - Noticeable edge blurring after median filtering



Idea of Improving Median Filtering

- Can we get rid of impulse noise without affecting clean pixels?
  - Yes, if we know where the clean pixels are

  or equivalently where the noisy pixels are
- How to detect noisy pixels?
  - They are black or white dots

Median Filtering with Noise Detection

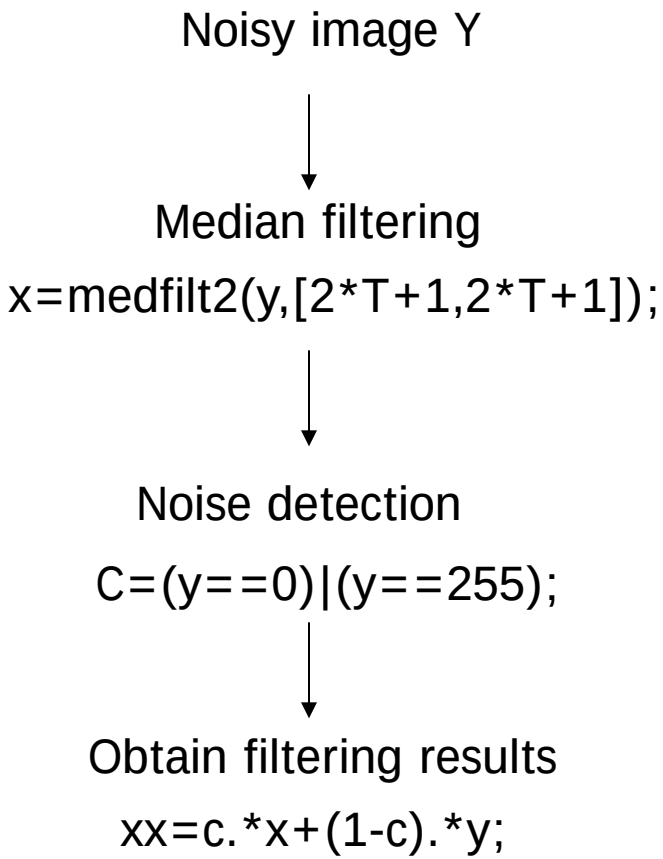
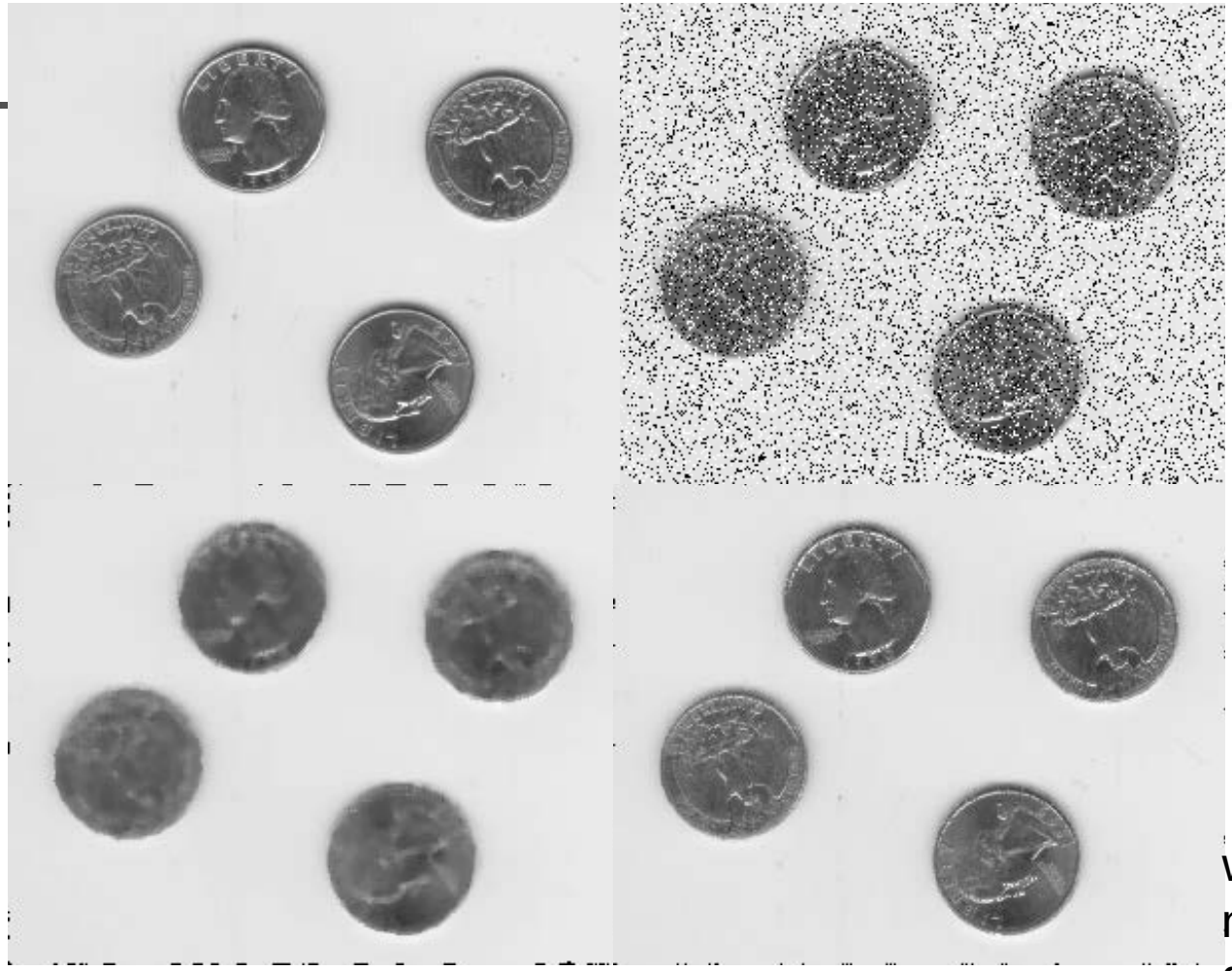


Image Example

clean



noisy
(p=0.2)

w/o
noise
detection

with
noise
detection

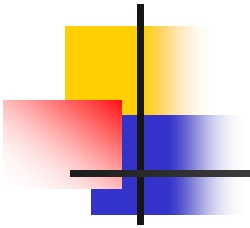


Image Denoising

- Introduction
- Impulse noise removal
  - Median filtering
- Additive white Gaussian noise removal
  - 2D convolution and DFT
- Periodic noise removal
  - Band-rejection and Notch filter

Additive White Gaussian Noise

Definition

Each pixel in an image is disturbed by a Gaussian random variable
With zero mean and variance σ^2

$$Y(i,j) = X(i,j) + N(i,j),$$

$$N(i,j) \sim N(0,\sigma^2), 1 \le i \le H, 1 \le j \le W$$

X: noise-free image, Y: noisy image

Note: unlike impulse noise situation, every pixel in the image contaminated by AWGN is noisy

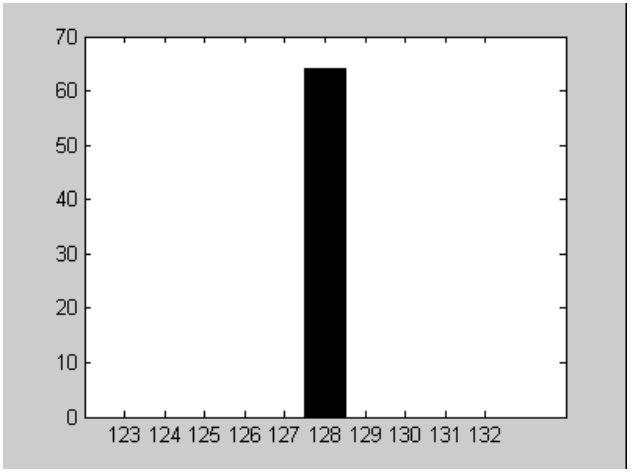
Numerical Example

128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128

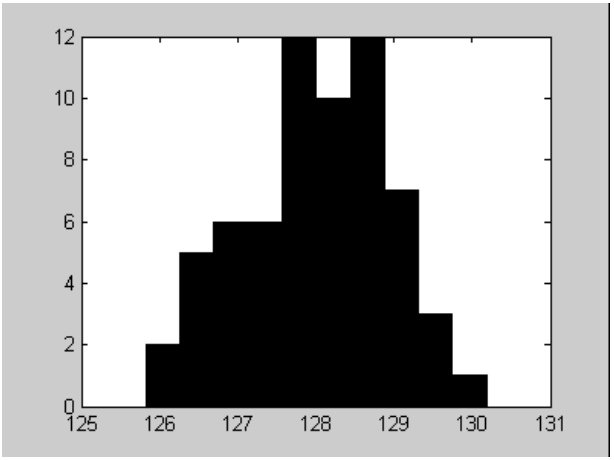
$\sigma^2 = 1$

128	128	129	127	129	126	126	128
126	128	128	129	129	128	128	127
128	128	128	129	129	127	127	128
128	129	127	126	129	129	129	128
127	127	128	127	129	127	129	128
129	130	127	129	127	129	130	128
129	128	129	128	128	128	129	129
128	128	130	129	128	127	127	126

X



Y

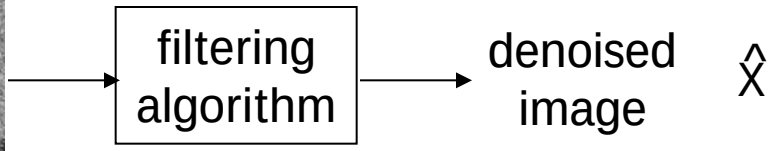


MATLAB Command

```
>Y = IMNOISE(X,'gaussian',m,v)
```

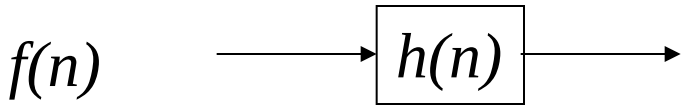
Image Denoising

Noisy image Y



Question: Why not use median filtering?
Hint: the noise type has changed.

1D Linear Filtering



See review section

$$g(n) = \sum_{k=-\infty}^{\infty} h(k) f(n-k) = h(n) \otimes f(n) = f(n) \otimes h(n)$$

Linear convolution

- Linearity $a_1 f_1(n) + a_2 f_2(n) \rightarrow a_1 g_1(n) + a_2 g_2(n)$

- Time-invariant property $f(n-n_0) \rightarrow g(n-n_0)$

Fourier Series

forward $$F(w) = \sum_{-\infty}^{\infty} f(n) e^{-jwn}$$

$$\updownarrow$$

inverse $$f(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(w) e^{jwn} dw$$

time-domain convolution $f(n) \otimes h(n)$

frequency-domain multiplication $F(w)H(w)$

Note that the input signal is a discrete sequence while its FT is a continuous function

Filters Examples

A **low-pass filter** is a filter that passes signals with a frequency lower than a certain cutoff frequency and attenuates signals with frequencies higher than the cutoff frequency. The amount of attenuation for each frequency depends on the filter design. The filter is sometimes called a **high-cut filter**

A **high-pass filter** is an electronic filter that passes signals with a frequency higher than a certain cutoff frequency and attenuates signals with frequencies lower than the cutoff frequency. The amount of attenuation for each frequency depends on the filter design. It is sometimes called a **low-cut filter**

Filter Examples

Low-pass (LP)

h(n)=[1,1]

↓

|h(w)|=2cos(w/2)

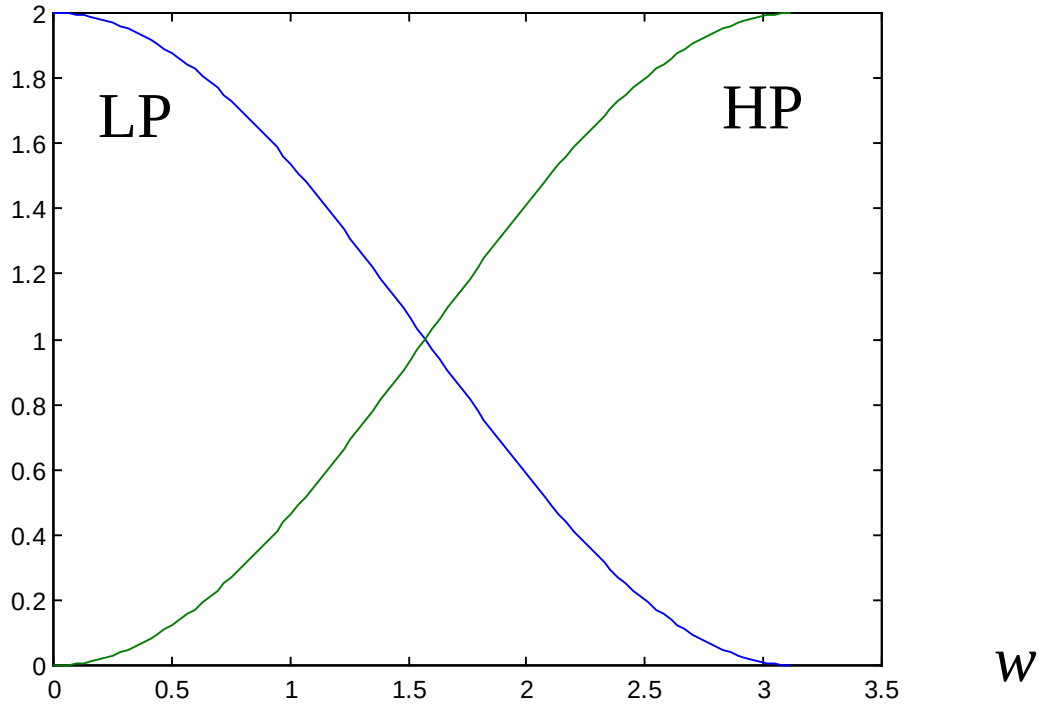
High-pass (LP)

h(n)=[1,-1]

↓

|h(w)|=2sin(w/2)

$|H(w)|$



1D Discrete Fourier Transform

forward transform $\qquad\qquad$ inverse transform

$$y(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn} \longleftrightarrow x(n)=\sum_{n=0}^{N-1}y(k)W_N^{-kn}$$

- Properties

  - periodic $\quad y(k+N)=y(k)$

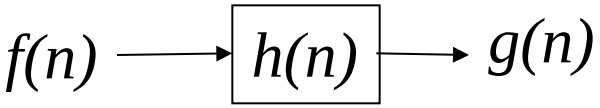
$$W_N=\exp\{-\frac{j2\pi}{N}\}$$

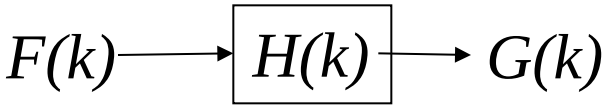
  - conjugate symmetric $\quad y(N-k)=y^*(k)$

Fast Fourier Transform (FFT)*

- Invented by Tukey and Cooley in 1965
- Basic idea: divide-and-conquer
- Reduce the complexity of N-point DFT from $O(N^2)$ to $O(N\log_2 N)$

Filtering in the Frequency Domain

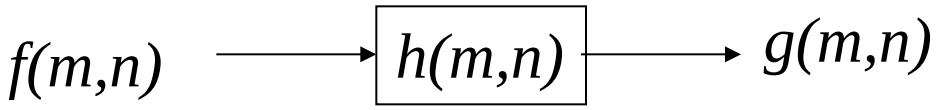




$$g(n) = f(n) \otimes h(n) \xrightarrow{\text{DFT}} G(k) = F(k)H(k)$$

convolution in the time domain is equivalent to multiplication in the frequency domain

2D Linear Filtering

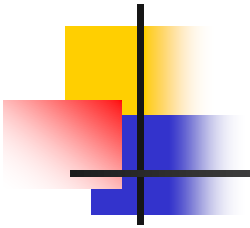


$$g(m,n) = \sum_{k,l=-\infty}^{\infty} h(k,l)\, f(m-k, n-l) = h(m,n) \otimes f(m,n)$$

2D convolution

MATLAB function: C = CONV2(A, B)

2D Filtering=Two Sequential 1D Filtering



- Just as we have observed with 2D transform, 2D (separable) filtering can be viewed as two sequential 1D filtering operations: one along row direction and the other along column direction
- The order of filtering does not matter

$$h(m,n) = h^1(m) \otimes h^1(n) = h^1(n) \otimes h^1(m)$$

h^1 : 1D filter

Numerical Example

1D filter $h^1(m)=[1,1]$, $h^1(n)=[1,-1]$

$$h^1(m) \otimes h^1(n) = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \qquad h^1(n) \otimes h^1(m) = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}$$

MATLAB command:

```
>h1=[1,1];h2=[1,-1];
>conv2(h1,h2)
>conv2(h2,h1)
```

Fourier Series (2D case)

$$F(w_1, w_2) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m,n) e^{-j(w_1 m + w_2 n)}$$

spatial-domain convolution

$$f(m,n) \otimes h(m,n)$$

frequency-domain multiplication

$$F(w_1, w_2) H(w_1, w_2)$$

Note that the input signal is discrete while its FT is a continuous function

Filter Examples

Low-pass (LP)

1D

$h^1(n)=[1,1]$

$\downarrow$

$|h^1(w)|=2\cos(w/2)$

$\downarrow$

2D

$h(n)=[1,1;1,1]$

$\downarrow$

$|h(w_1,w_2)|=4\cos(w_1/2)\cos(w_2/2)$

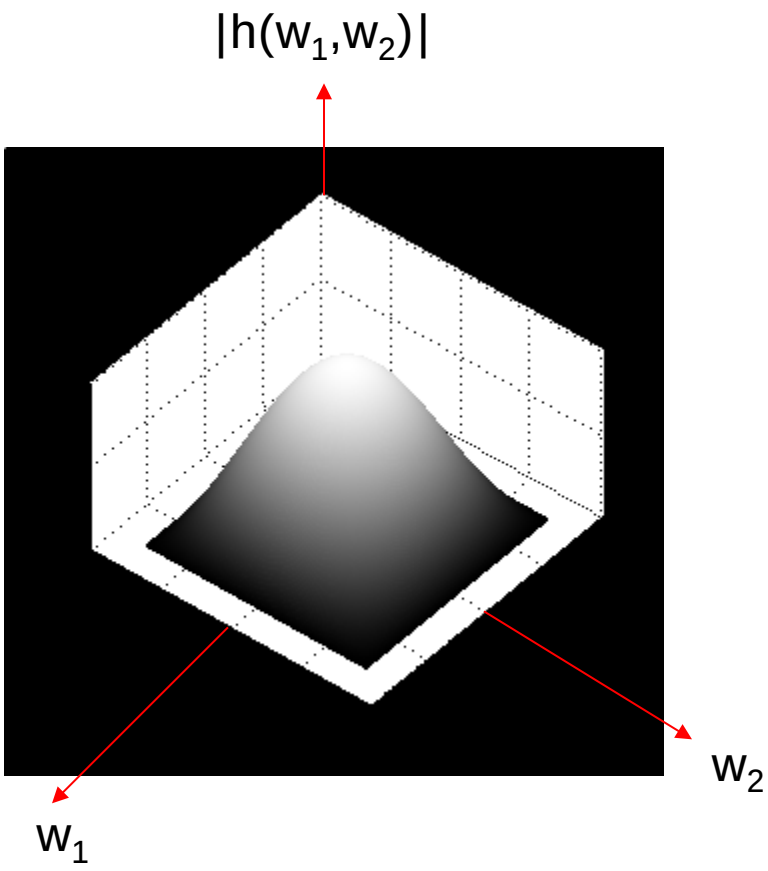
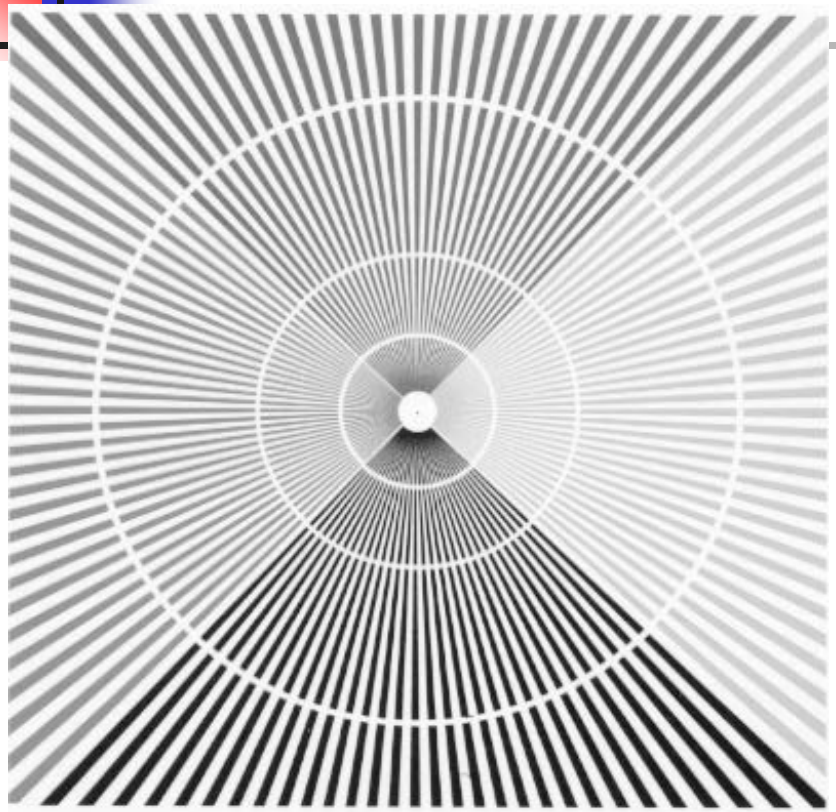
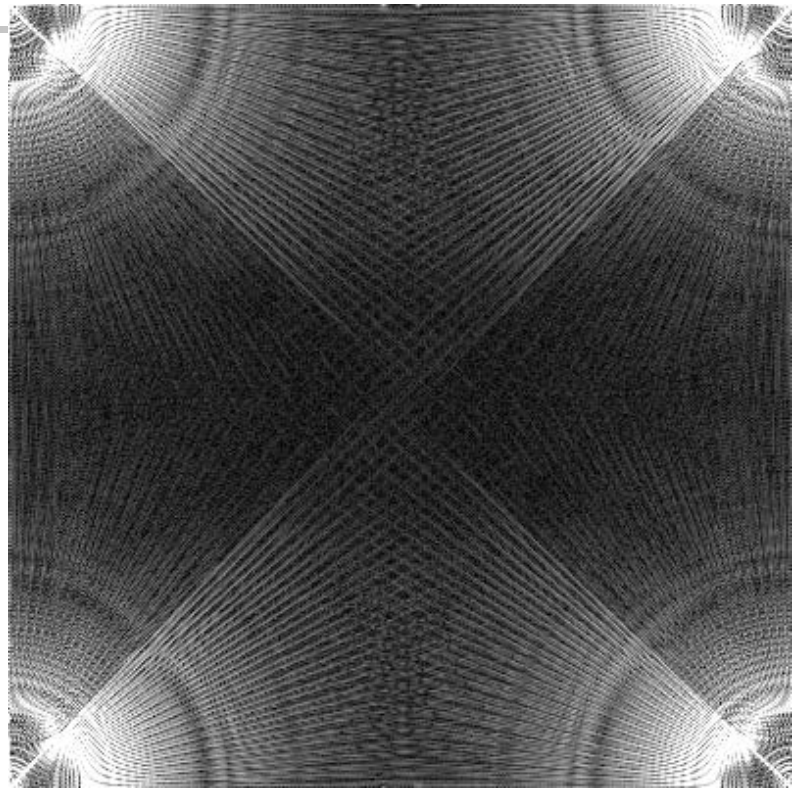


Image DFT Example

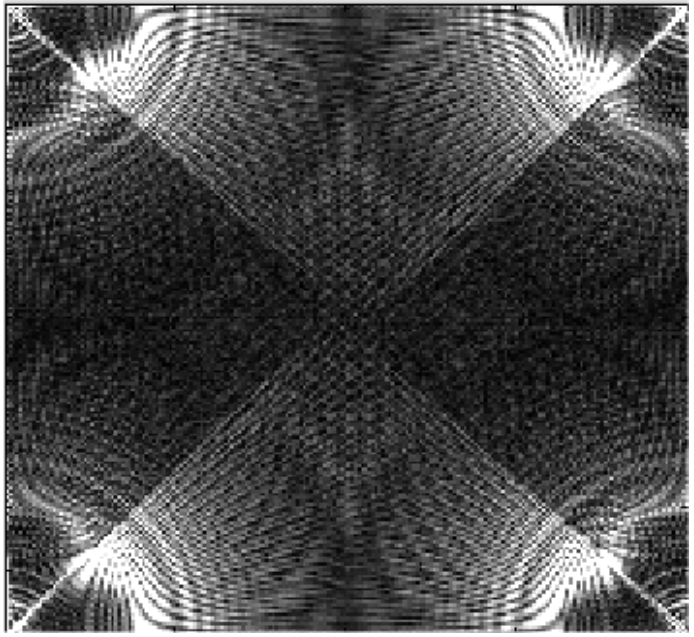


Original ray image **X**

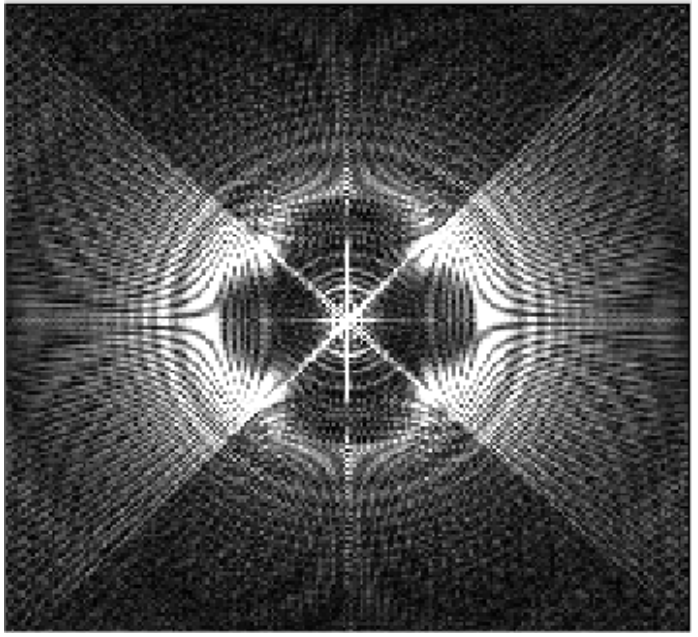


choice 1: **Y**=fft2(**X**)

Image DFT Example (Con't)



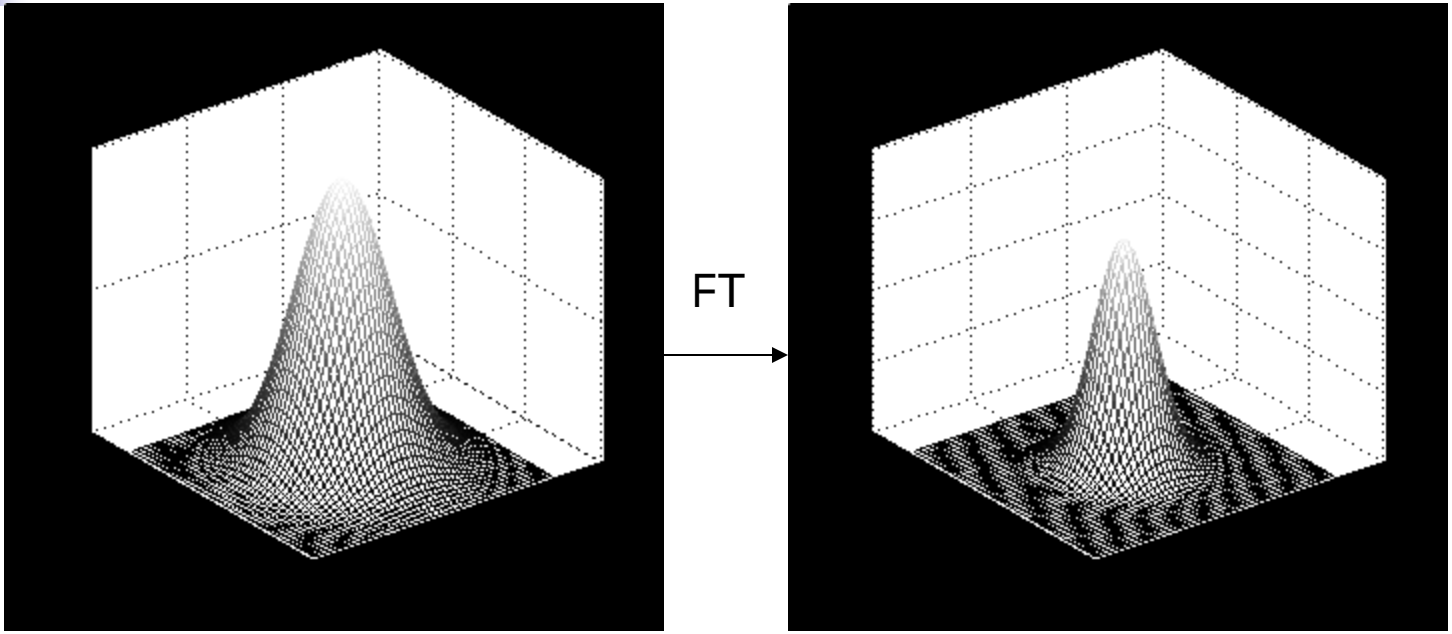
choice 1: **Y**=fft2(**X**)
Low-frequency at four corners



choice 2: **Y**=fftshift(fft2(**X**))
Low-frequency at the center

FFTSHIFT Shift zero-frequency component to center of spectrum.

Gaussian Filter



$$h(m,n) = \exp(-\frac{m^2+n^2}{2\sigma^2})$$

$$H(w_1,w_2) = \exp(-\frac{w_1^2+w_2^2}{2\sigma^2})$$

MATLAB code: >h=fspecial('gaussian', HSIZE,SIGMA);

Image Example

noisy

(σ=25)

denoised

(σ=1)

denoised

(σ=1.5)

Matlab functions: imfilter, filter2

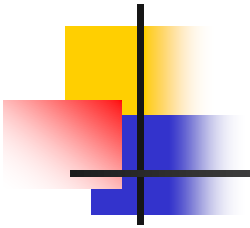
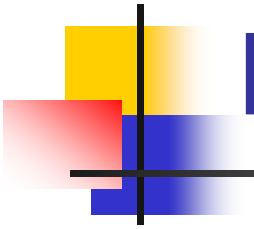


Image Denoising

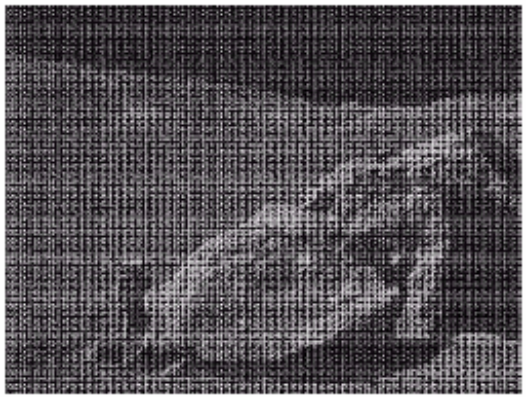
- Introduction
- Impulse noise removal
  - Median filtering
- Additive white Gaussian noise removal
  - 2D convolution and DFT
- Periodic noise removal
  - Band-rejection and Notch filter



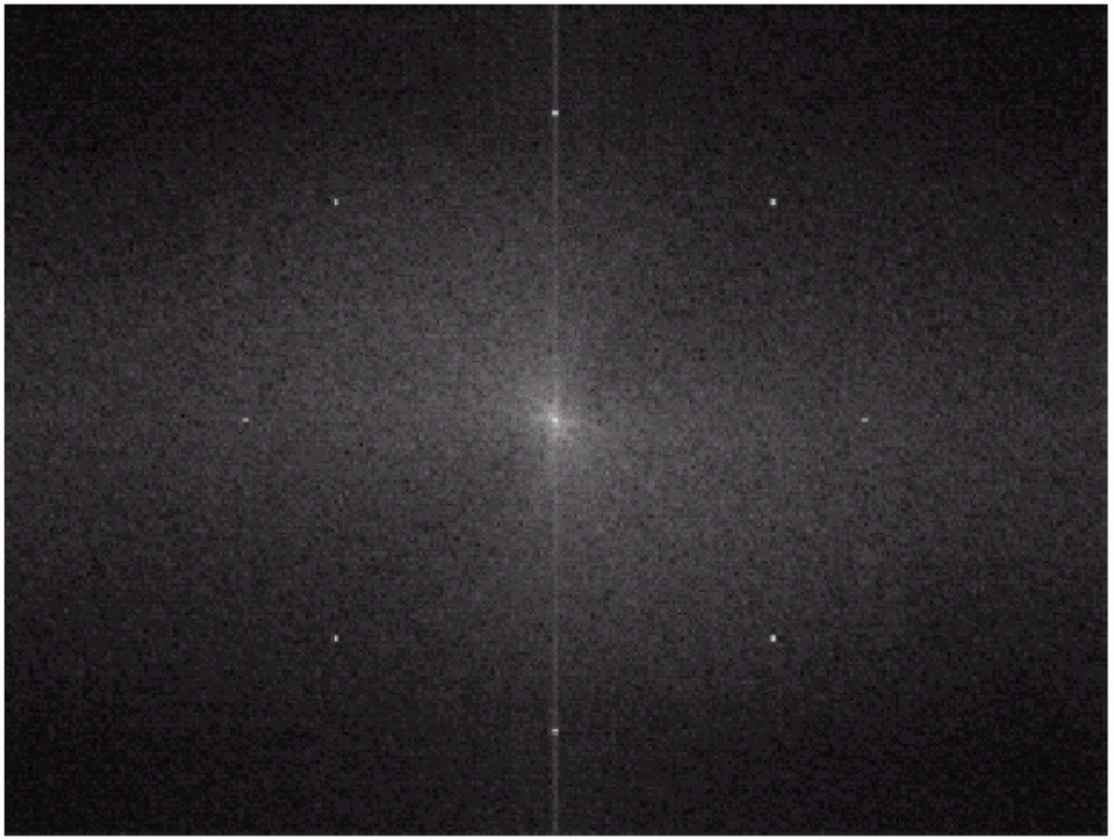
Periodic Noise

- Source: electrical or electromechanical interference during image acquistion
- Characteristics
  - Spatially dependent
  - Periodic – easy to observe in frequency domain
- Processing method
  - Suppressing noise component in frequency domain

Image Example



spatial

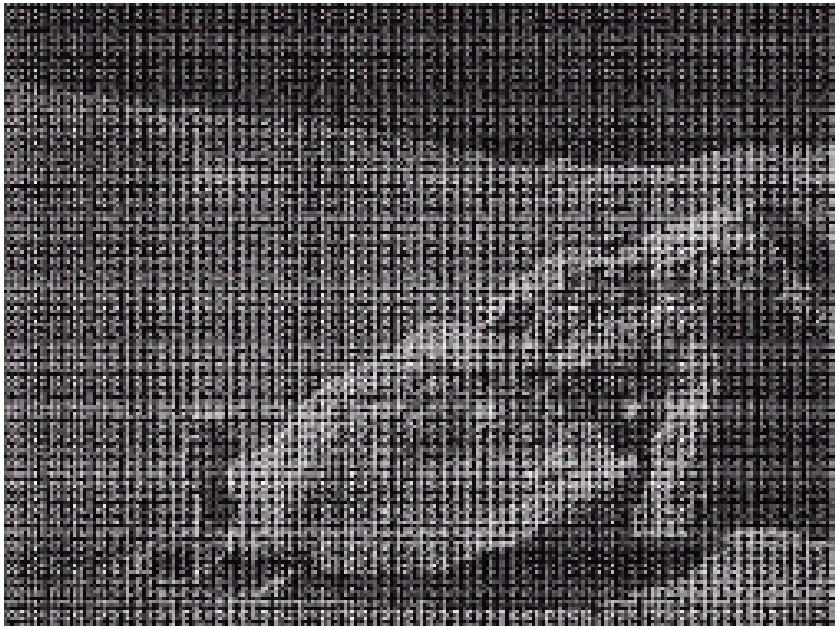


Frequency (note the four pairs of bright dots)

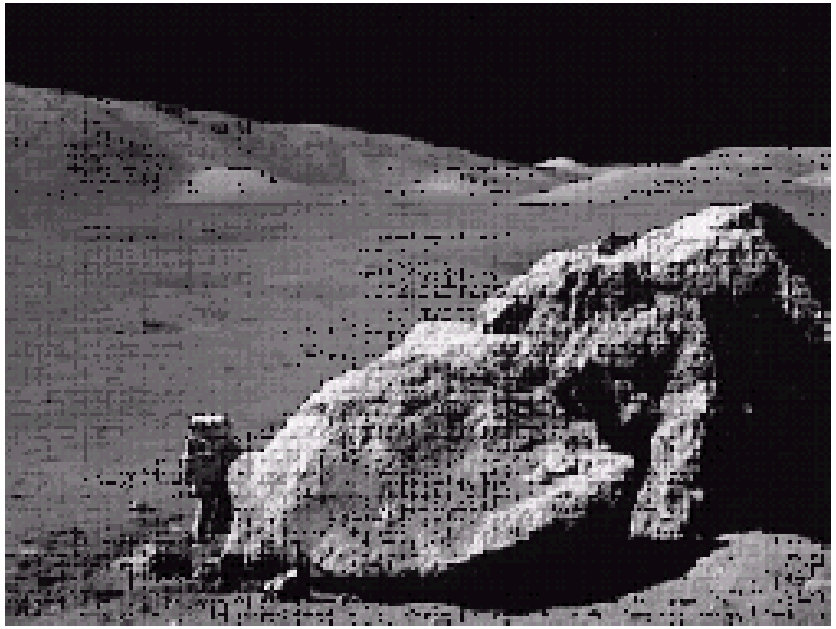
Band Rejection Filter

In signal processing, a **band-stop filter** or **band-rejection filter or notch filter** is a filter that passes most frequencies unaltered, but attenuates those in a specific range to very low levels. It is the opposite of a band-pass filter.

Image Example



Before filtering



After filtering