Digital Image Processing

Image Restoration

Image Restoration

- Image restoration vs. image enhancement
  - Enhancement:
    - largely a subjective process
    - Priori knowledge about the degradation is not a must (sometimes no degradation is involved)
    - Procedures are heuristic and take advantage of the psychophysical aspects of human visual system
  - Restoration:
    - more an objective process
    - Images are degraded
    - Tries to recover the images by using the knowledge about the degradation

An Image Degradation Model

- Two types of degradation
  - Additive noise
    - Spatial domain restoration (denoising) techniques are preferred
  - Image blur
    - Frequency domain methods are preferred
- We model the degradation process by a degradation function h(x,y), an additive noise term, $\eta(x,y)$, as $g(x,y)=h(x,y)*f(x,y)+\eta(x,y)$
  - f(x,y) is the (input) image free from any degradation
  - g(x,y) is the degraded image
  - $*$ is the convolution operator
  - The goal is to obtain an estimate of f(x,y) according to the knowledge about the degradation function h and the additive noise η
  - In frequency domain: $G(u,v)=H(u,v)F(u,v)+N(u,v)$
- Three cases are considered here
  - $g(x,y)=f(x,y)+\eta(x,y)$
  - $g(x,y)=h(x,y)*f(x,y)$
  - $g(x,y)=h(x,y)*f(x,y)+\eta(x,y)$

A Model of the Image Degradation/Restoration Process

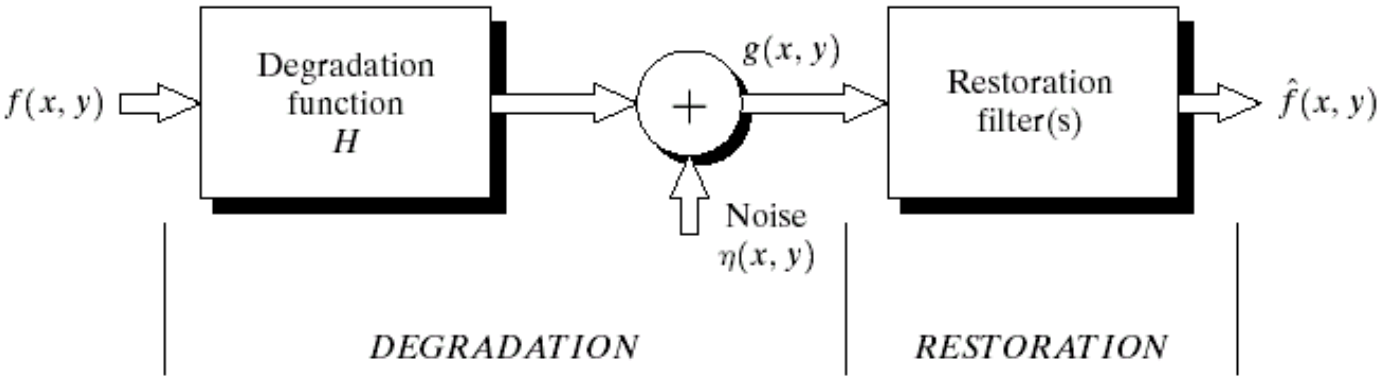


FIGURE 5.1 A model of the image degradation/restoration process.

Noise Model

- We first consider the degradation due to noise only
  - h is an impulse for now (H is a constant)
- White noise
  - Autocorrelation function is an impulse function multiplied by a constant
    - $$a(x,y) = \sum_{t=0}^{N-1}\sum_{s=0}^{M-1}\eta(s,t)\cdot\eta(s-x,t-y) = N_0\delta(x,y)$$
    - It means there is no correlation between any two pixels in the noise image
    - There is no way to predict the next noise value
  - The spectrum of the autocorrelation function is a constant (white)

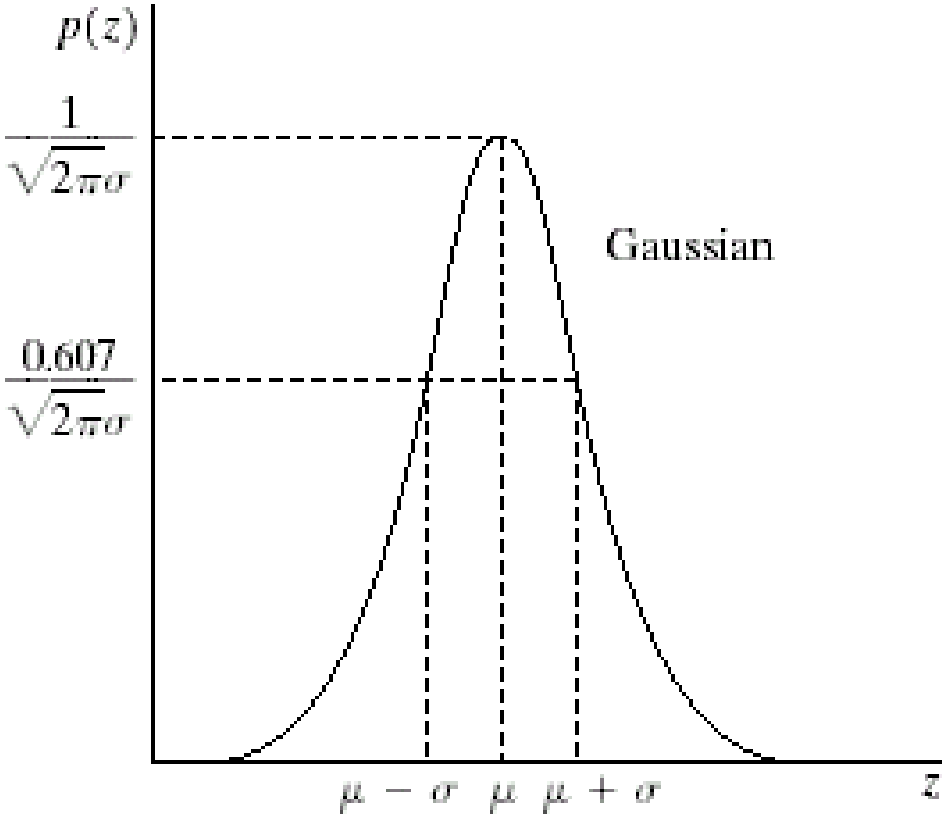
Gaussian Noise

- Noise (image) can be classified according the distribution of the values of pixels (of the noise image) or its (normalized) histogram
- Gaussian noise is characterized by two parameters, μ (mean) and σ^2 (variance), by

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(z-\mu)^2/2\sigma^2}$$

- 70% values of z fall in the range $[(\mu-\sigma),(\mu+\sigma)]$
- 95% values of z fall in the range $[(\mu-2\sigma),(\mu+2\sigma)]$

Gaussian Noise



Other Noise Models

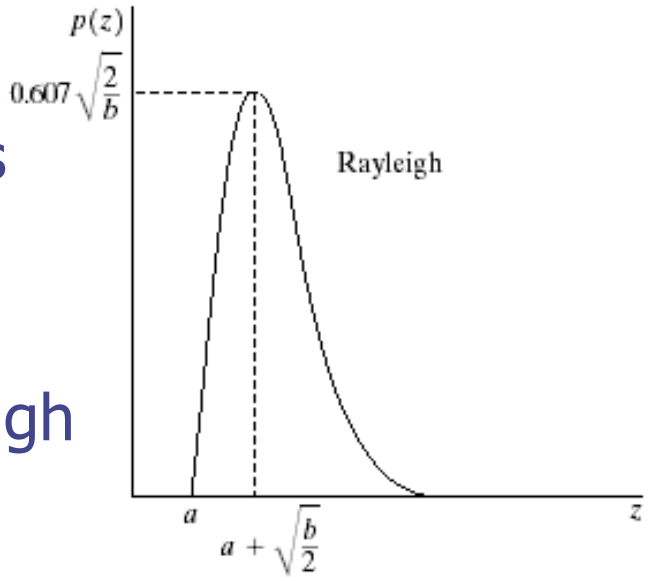
- **Rayleigh** noise

$$p(z) = \begin{cases} \frac{2}{b}(z-a)e^{-(z-a)^2/b} & \text{for } z \geq a \\ 0 & \text{for } z < a \end{cases}$$

  - The mean and variance of this density are given by

$$\mu = a + \sqrt{\pi b/4} \quad \text{and} \quad \sigma^2 = \frac{b(4-\pi)}{4}$$

  - a and b can be obtained through mean and variance



Other Noise Models

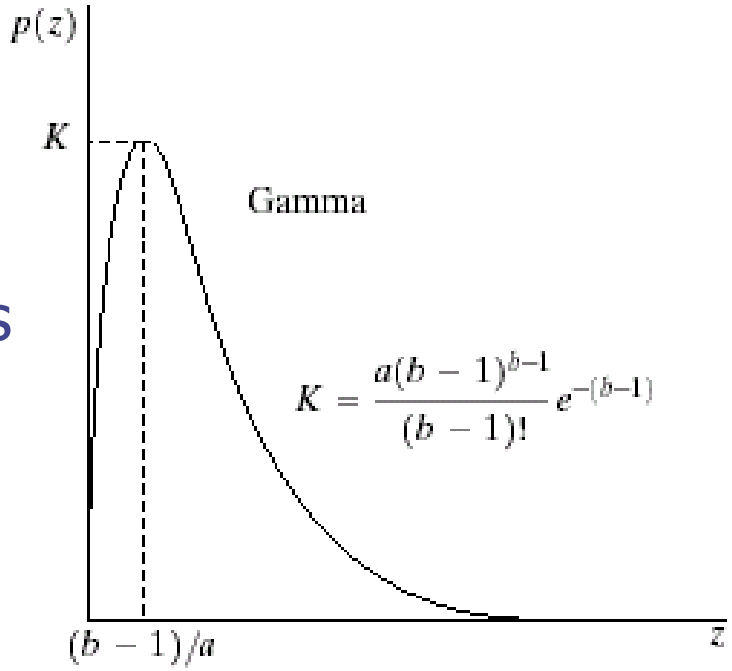
- **Erlang (Gamma)** noise

$$p(z) = \begin{cases} \dfrac{a^b z^{b-1}}{(b-1)!} e^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases}$$

  - The mean and variance of this density are given by

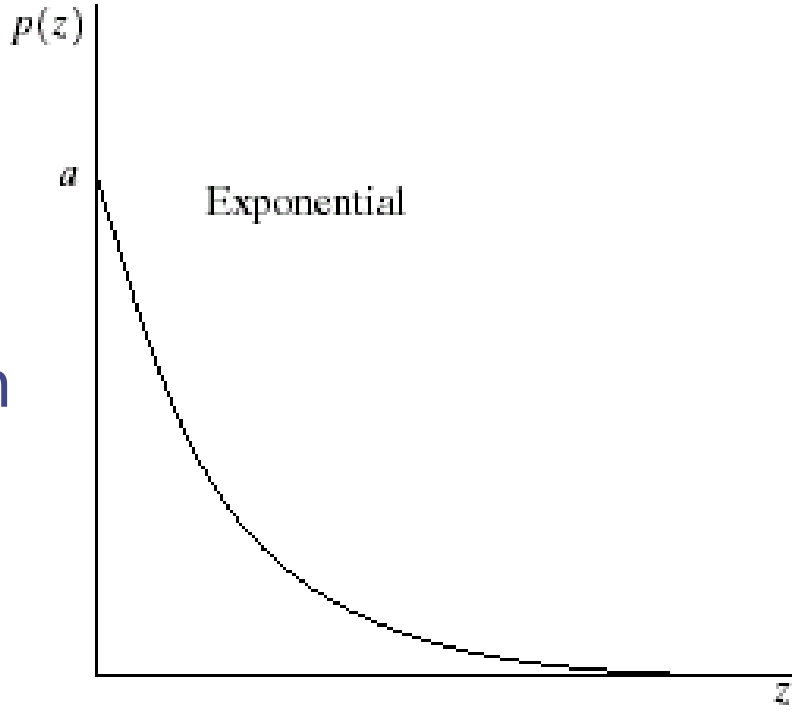
$$\mu = b/a \quad \text{and} \quad \sigma^2 = \frac{b}{a^2}$$

  - a and b can be obtained through mean and variance



Other Noise Models

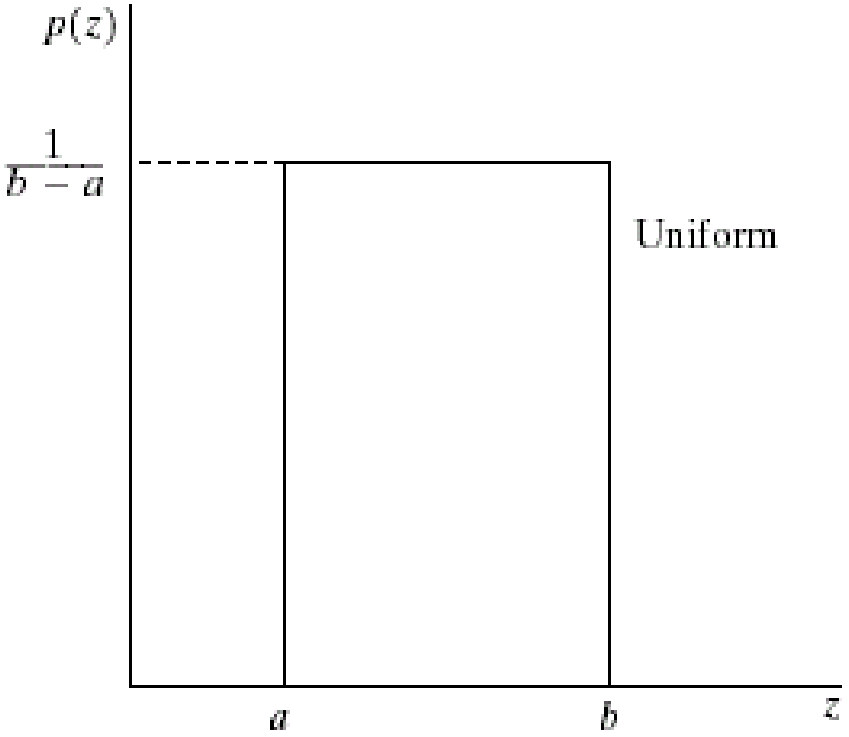
- **Exponential** noise
  - $$p(z) = \begin{cases} ae^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases}$$
  - The mean and variance of this density are given by
    $$\mu = 1/a \text{ and } \sigma^2 = \frac{1}{a^2}$$
  - Special case pf Erlang PDF with b=1



Other Noise Models

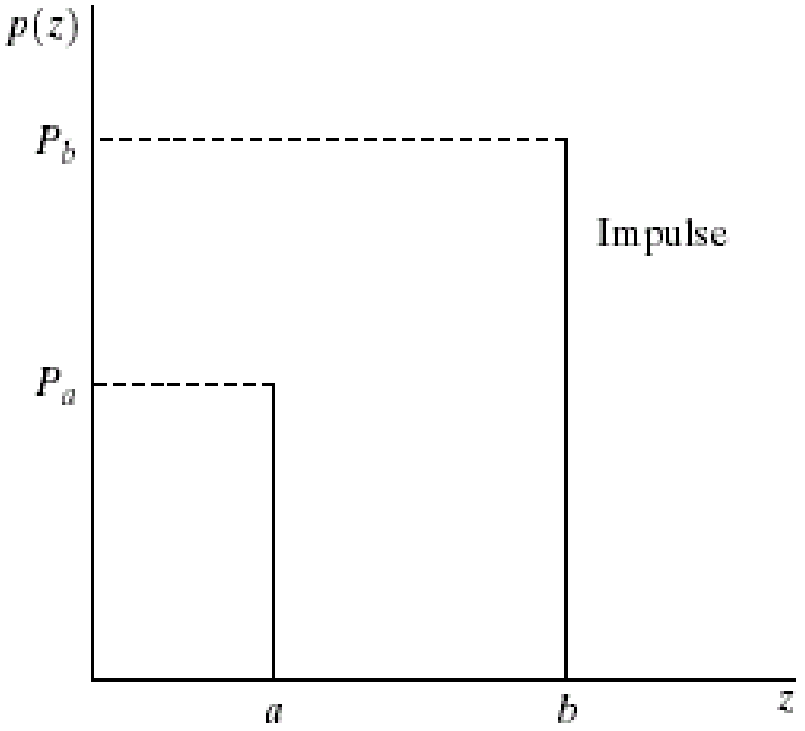
- **Uniform** noise
  - $$p(z) = \begin{cases} \frac{1}{b-a} & \text{if } a \le z \le b \\ 0 & \text{otherwise} \end{cases}$$
  - The mean and variance of this density are given by

$$\mu = (a+b)/2 \text{ and } \sigma^2 = \frac{(b-a)^2}{12}$$



Other Noise Models

- **Impulse** (salt-and-pepper) noise
  - $$p(z) = \begin{cases} P_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases}$$
    - If either Pa or Pb is zero, the impulse noise is called unipolar
    - a and b usually are extreme values because impulse corruption is usually large compared with the strength of the image signal
    - It is the only type of noise that can be distinguished from others visually



The Principle Use of Noise Models

- **Gaussian noise:**
  - electronic circuit noise and sensor noise due to poor illumination or high temperature.
- **Rayleigh noise:**
  - Noise in range imaging.
- **Erlang noise:**
  - Noise in laser imaging.
- **Impulse noise:**
  - Quick transients take place during imaging.
- **Uniform noise:**
  - Used in simulations.

A Sample Image

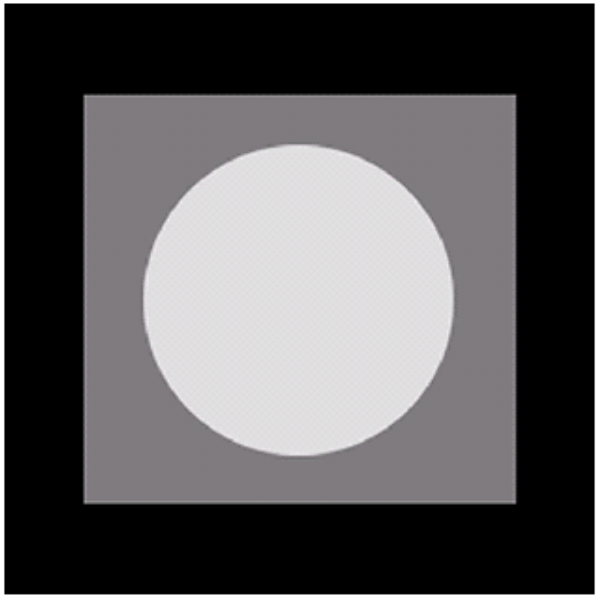
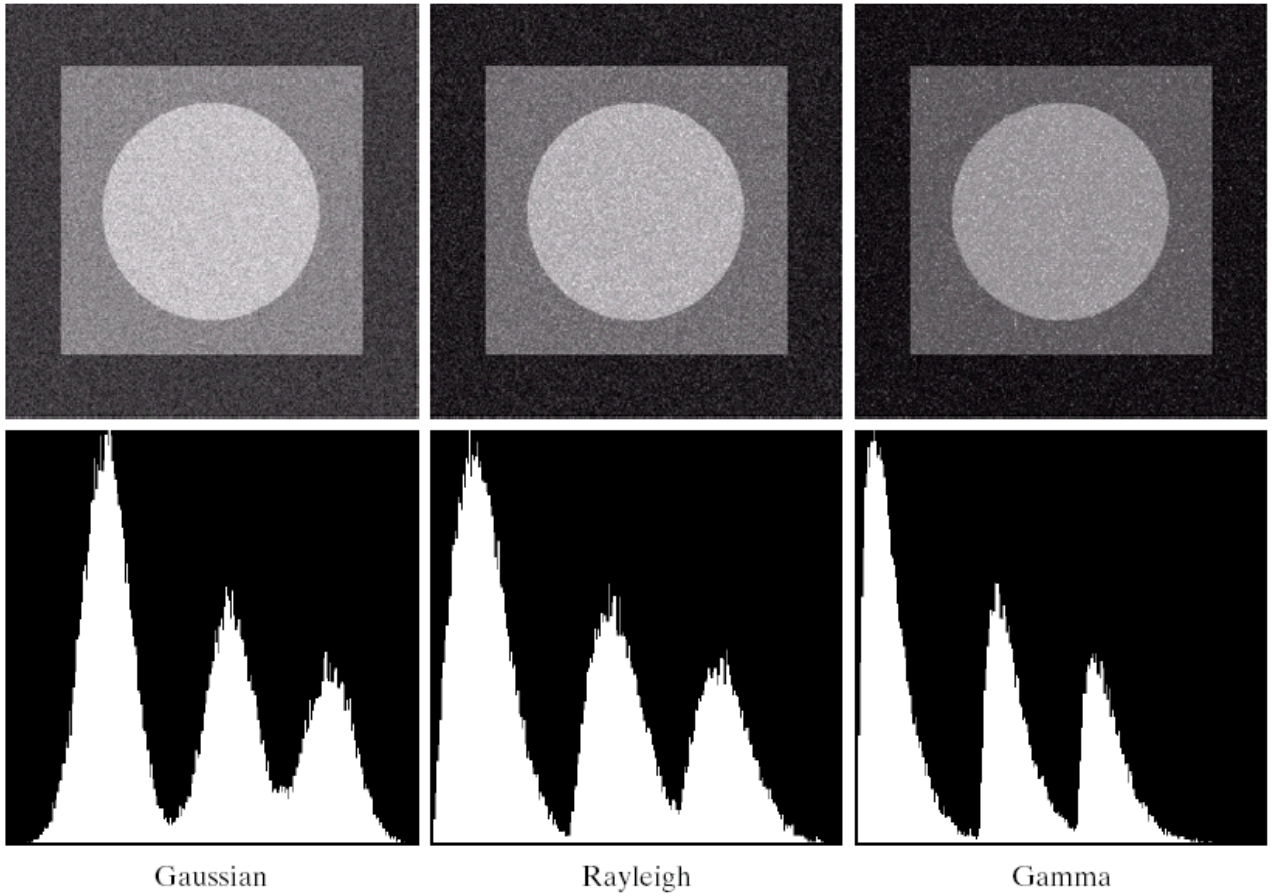


FIGURE 5.3 Test pattern used to illustrate the characteristics of the noise PDFs shown in Fig. 5.2.

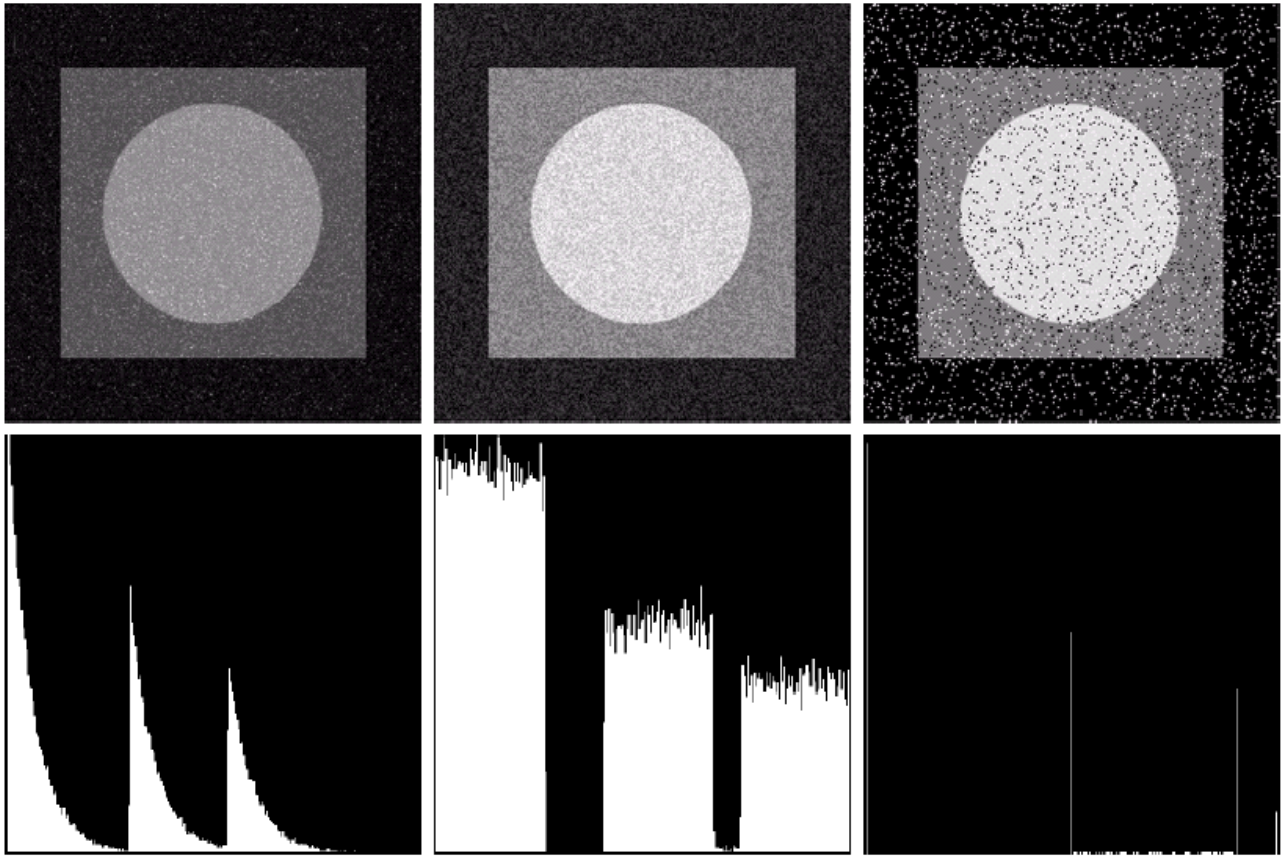
Effect of Adding Noise to Sample Image



a b c
d e f

FIGURE 5.4 Images and histograms resulting from adding Gaussian, Rayleigh, and gamma noise to the image in Fig. 5.3.

Effect of Adding Noise to Sample Image

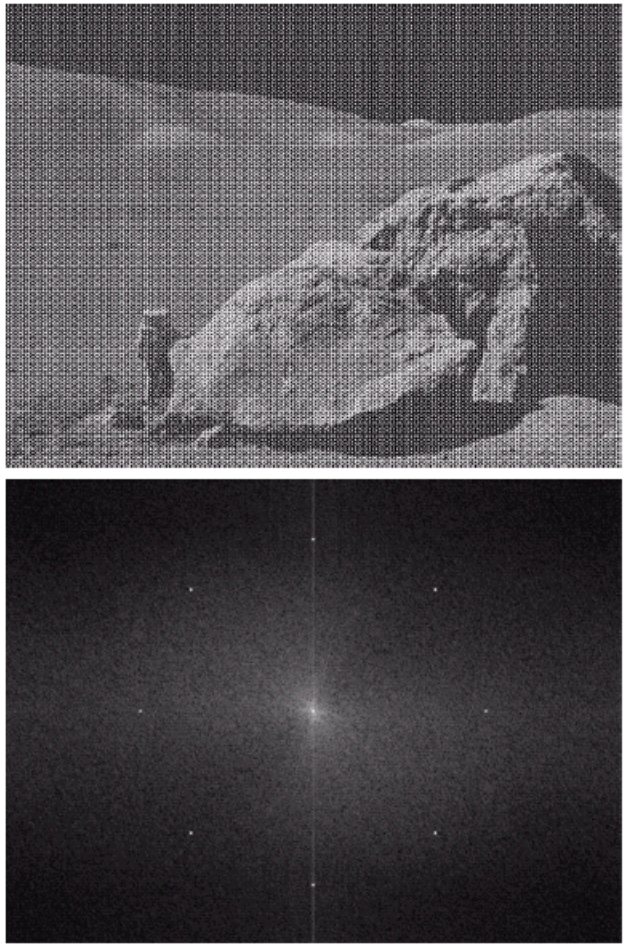


g h i
j k l

FIGURE 5.4 *(Continued)* Images and histograms resulting from adding exponential, uniform, and impulse noise to the image in Fig. 5.3.

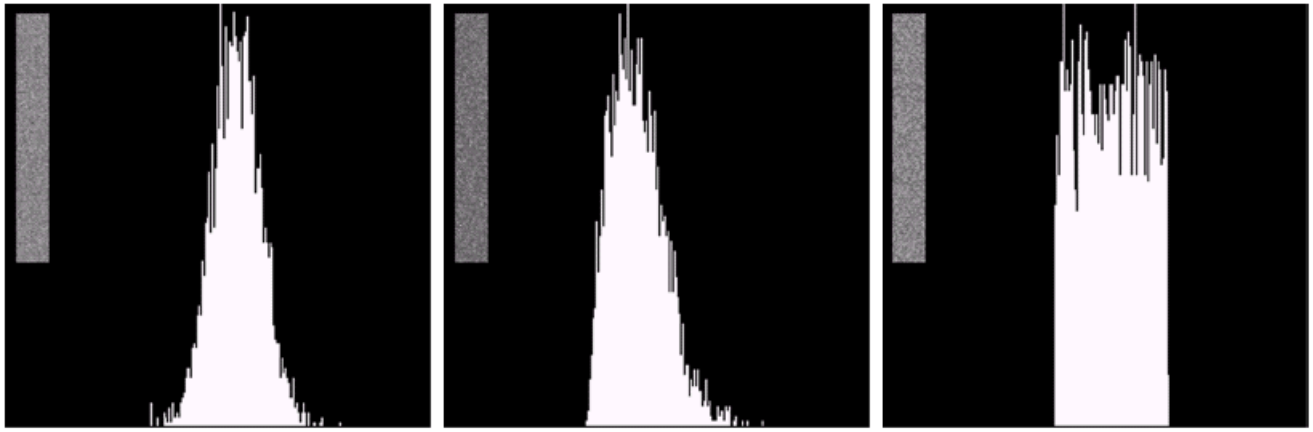
Periodic Noise

- Arises typically from electrical or electromechanical interference during image acquisition
- It can be observed by visual inspection both in the spatial domain and frequency domain
- The only spatially dependent noise will be considered



Estimation of Noise Parameters

- Periodic noise
  - Parameters can be estimated by inspection of the spectrum
- Noise PDFs
  - From sensor specifications
  - If imaging sensors are available, capture a set of images of plain environments
  - If only noisy images are available, parameters of the PDF involved can be estimated from small patches of constant regions of the noisy images



a b c

FIGURE 5.6 Histograms computed using small strips (shown as inserts) from (a) the Gaussian, (b) the Rayleigh, and (c) the uniform noisy images in Fig. 5.4.

Estimation of Noise Parameters

- In most cases, only mean and variance are to be estimated
  - Others can be obtained from the estimated mean and variance
- Assume a sub-image with plain scene is available and is denoted by S
  - $\hat{\mu} = \frac{1}{N_S} \sum_{(x_i, y_i) \in S} z(x_i, y_i)$
  - $\hat{\sigma}_1^{\,2} = \frac{1}{N_S} \sum_{(x_i, y_i) \in S} (z(x_i, y_i) - \hat{\mu})^2$

Restoration in the Presence of Noise Only (De-Noising)

- Mean filters
  - Arithmetic mean filter
    - g(x,y) is the corrupted image
    - $S_{x,y}$ is the mask

$$\hat{f}(x,y) = \frac{1}{mn} \sum_{(s,t)\in S_{x,y}} g(s,t)$$

  - Geometric mean filters
    - Tends to preserve more details

$$\hat{f}(x,y) = \left[\prod_{(s,t)\in S_{x,y}} g(s,t) \right]^{\frac{1}{mn}}$$

  - Harmonic mean filter

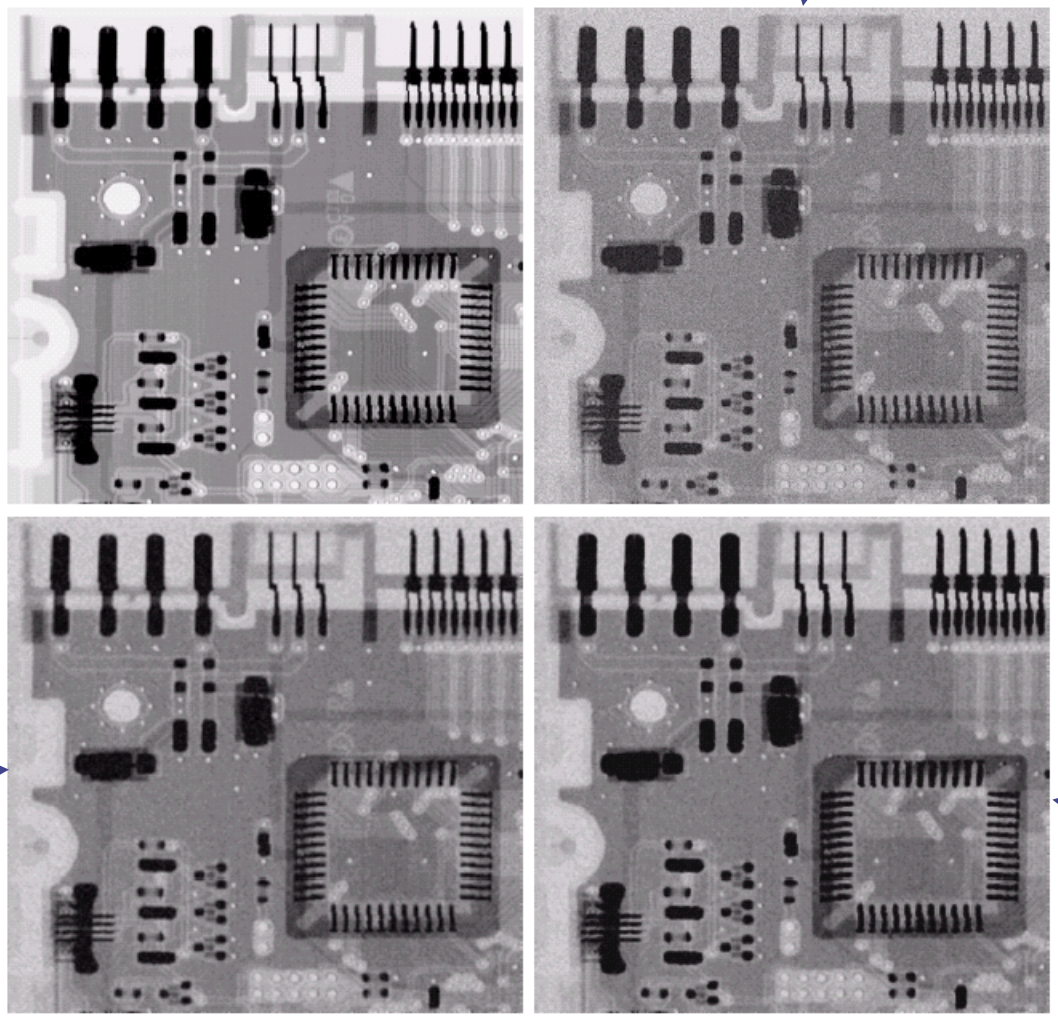
$$\hat{f}(x,y) = \frac{mn}{\sum_{(s,t)\in S_{x,y}} \frac{1}{g(s,t)}}$$

    - Works well for salt noise but fails for pepper noise
  - Contraharmonic mean filter
    - Q: order of the filter
    - Positive Q works for pepper noise
    - Negative Q works for salt noise
    - Q=0➔arithmetic mean filter
    - Q=-1➔harmonic mean filter

$$\hat{f}(x,y) = \frac{\sum_{(s,t)\in S_{x,y}} g(s,t)^{Q+1}}{\sum_{(s,t)\in S_{x,y}} g(s,t)^{Q}}$$

De-Noising

Corrupted by Gaussian Noise



a b
c d

FIGURE 5.7 (a) X-ray image. (b) Image corrupted by additive Gaussian noise. (c) Result of filtering with an arithmetic mean filter of size 3×3. (d) Result of filtering with a geometric mean filter of the same size. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

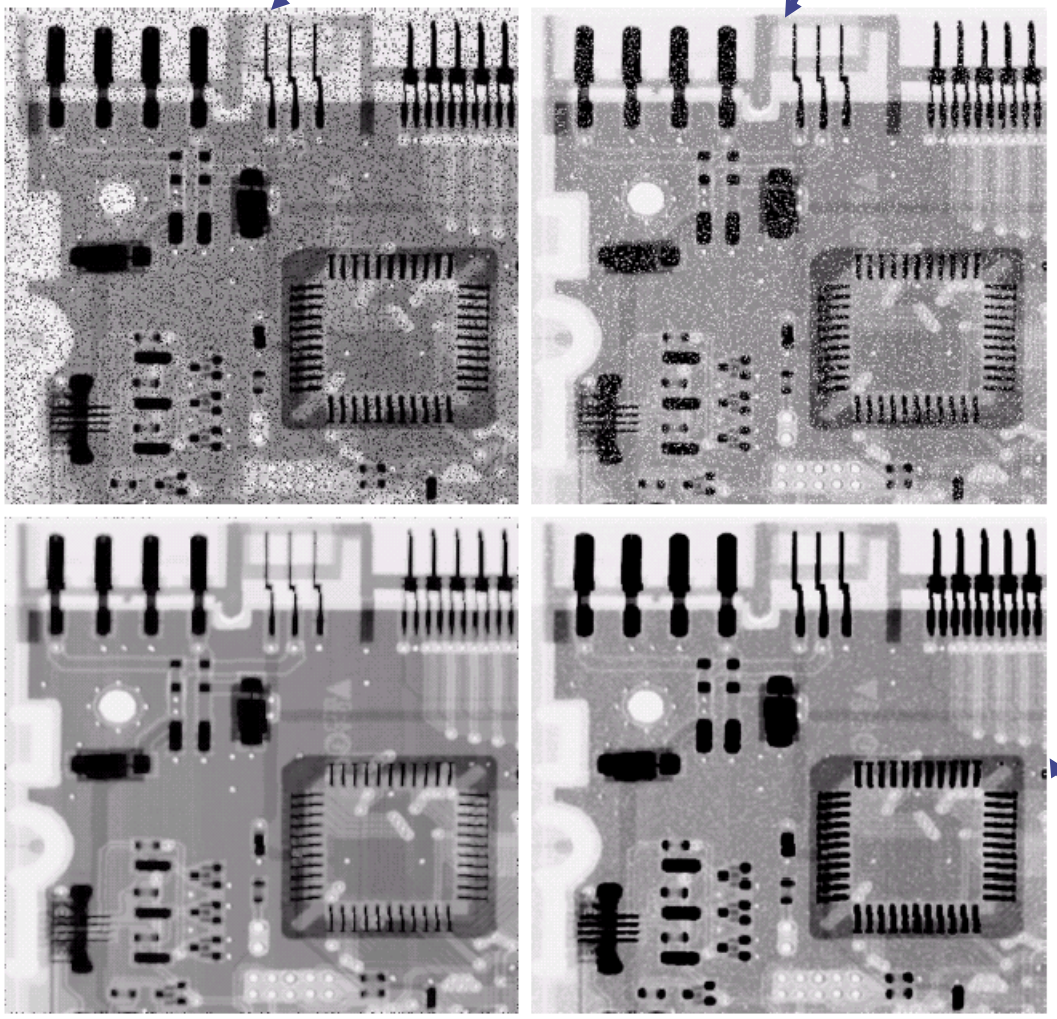
Mean Filtering

Geometric Mean Filtering

De-Noising

Corrupted by pepper noise

Corrupted by salt noise



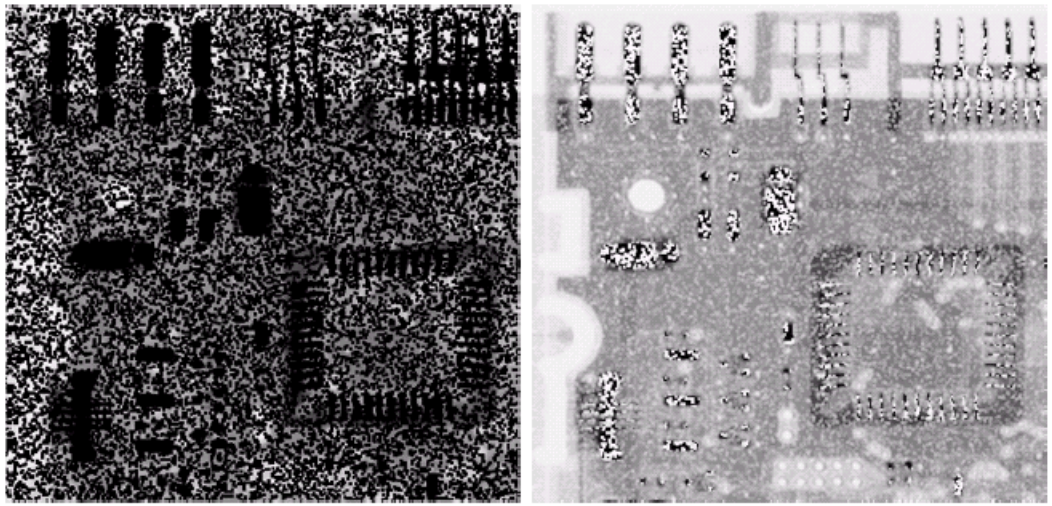
a b
c d

FIGURE 5.8
(a) Image corrupted by pepper noise with a probability of 0.1. (b) Image corrupted by salt noise with the same probability. (c) Result of filtering (a) with a 3×3 contraharmonic filter of order 1.5. (d) Result of filtering (b) with $Q = -1.5$.

3x3 Contraharmonic Q=1.5

3x3 Contraharmonic Q=-1.5

De-Noising



a b

FIGURE 5.9 Results of selecting the wrong sign in contraharmonic filtering. (a) Result of filtering Fig. 5.8(a) with a contraharmonic filter of size 3×3 and $Q = -1.5$. (b) Result of filtering 5.8(b) with $Q = 1.5$.

Filters Based on Order Statistics (De-Noising)

- Median filter

$$\hat{f}(x,y) = \underset{(s,t)\in S_{x,y}}{\text{median}}\{g(s,t)\}$$

  - Median represents the 50th percentile of a ranked set of numbers
- Max and min filter
  - Max filter uses the 100th percentile of a ranked set of numbers
    - Good for removing pepper noise
  - Min filter uses the 1 percentile of a ranked set of numbers
    - Good for removing salt noise
- Midpoint filter

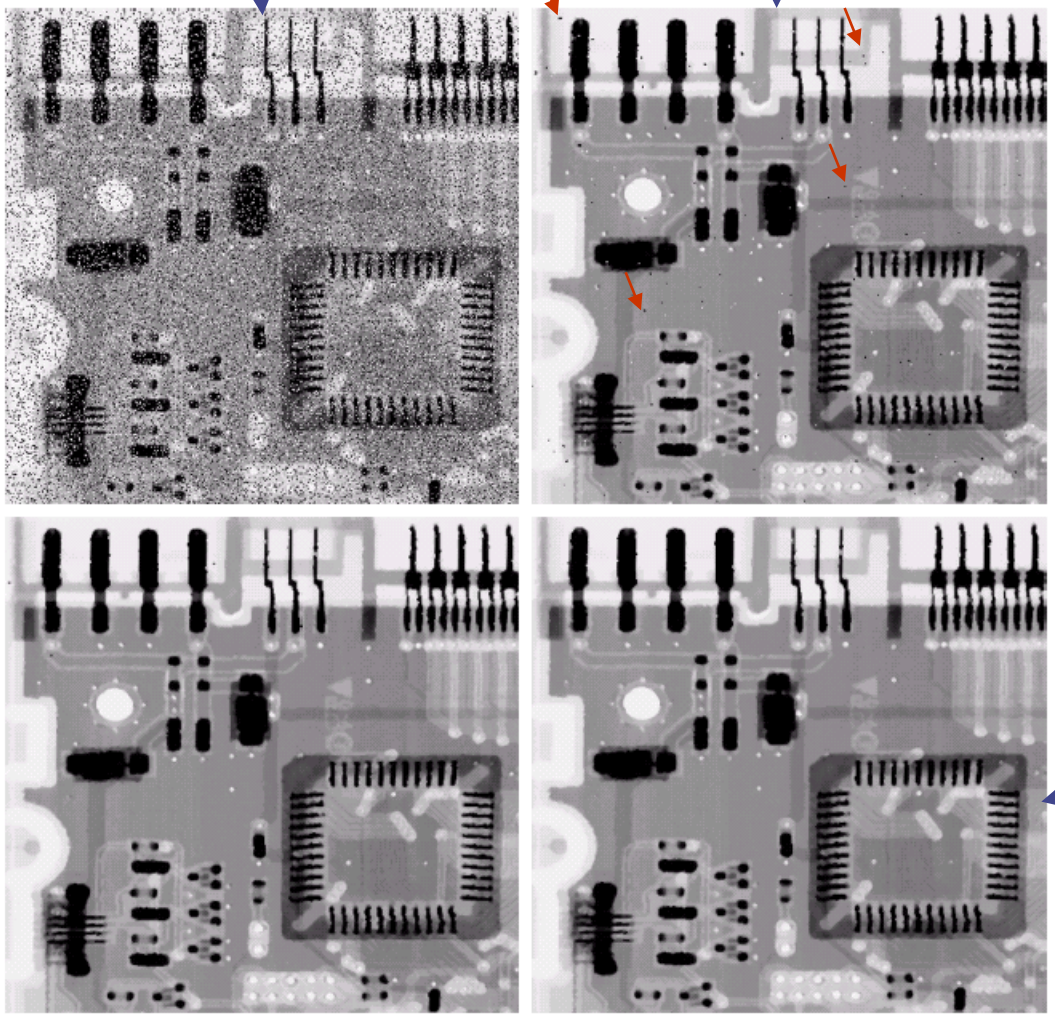
$$\hat{f}(x,y) = \frac{1}{2}\left[\max_{(s,t)\in S_{xy}}\{g(s,t)\} + \min_{(s,t)\in S_{xy}}\{g(s,t)\}\right]$$

  - Works best for noise with symmetric PDF like Gaussian or uniform noise

De-Noising

Corrupted by salt & pepper noise

One pass median filtering



a b
c d

FIGURE 5.10
(a) Image corrupted by salt-and-pepper noise with probabilities $P_a = P_b = 0.1$.
(b) Result of one pass with a median filter of size 3 × 3.
(c) Result of processing (b) with this filter.
(d) Result of processing (c) with the same filter.

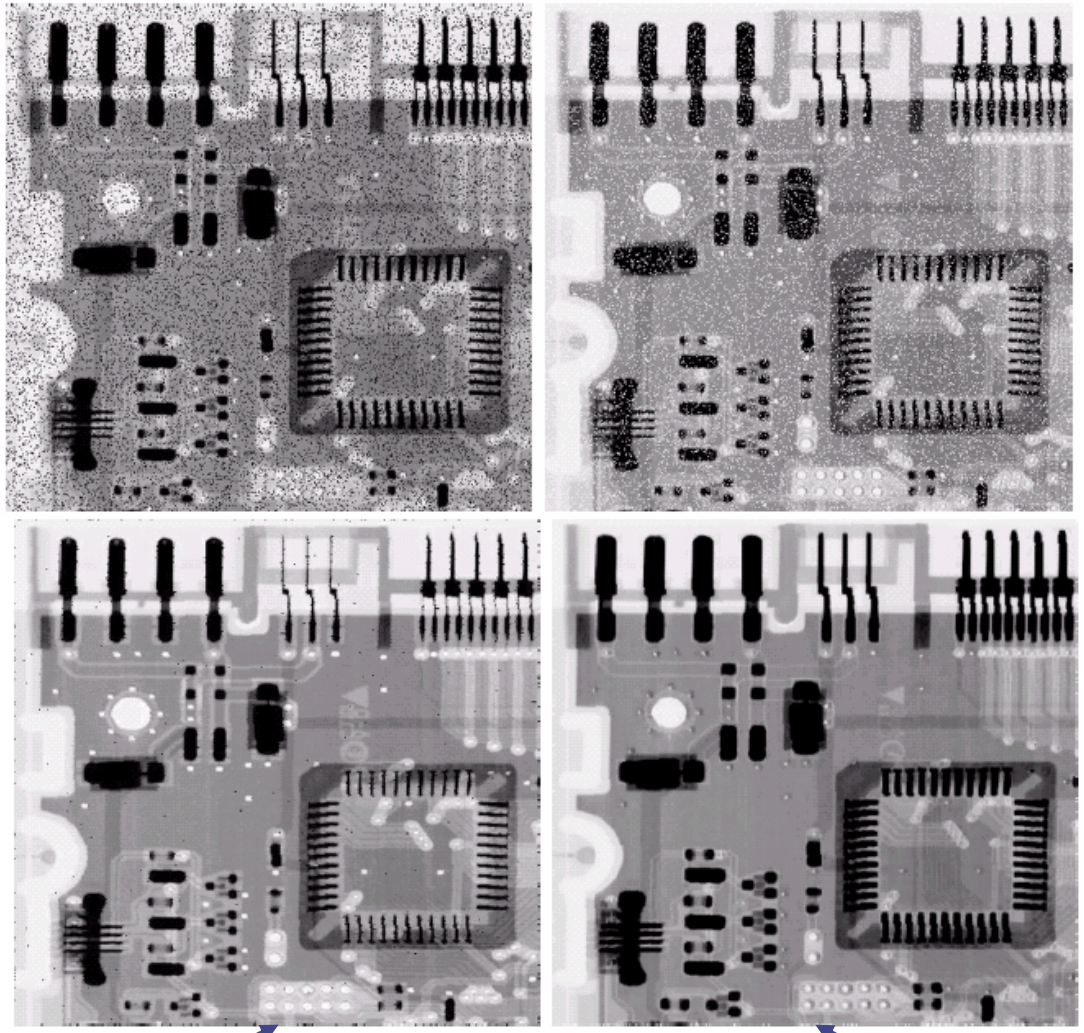
Two pass median filtering

Three pass median filtering

De-Noising

Corrupted by pepper noise

Corrupted by salt noise



Max Filtering

Min Filtering

a b

FIGURE 5.11 (a) Result of filtering Fig. 5.8(a) with a max filter of size 3×3. (b) Result of filtering 5.8(b) with a min filter of the same size.

Alpha-Trimmed Mean Filter (De-Noising)

- Alpha-trimmed mean filter takes the mean value of the pixels enclosed by an m×n mask after deleting the pixels with the d/2 lowest and the d/2 highest gray-level values

$$\hat{f}(x,y) = \frac{1}{mn-d} \sum_{(s,t)\in S_{xy}} g_r(s,t)$$

  - $g_r(s,t)$ represent the remaining mn-d pixels
  - It is useful in situations involving multiple types of noise like a combination of salt-and-pepper and Gaussian

De-Noising

Corrupted by additive Uniform noise

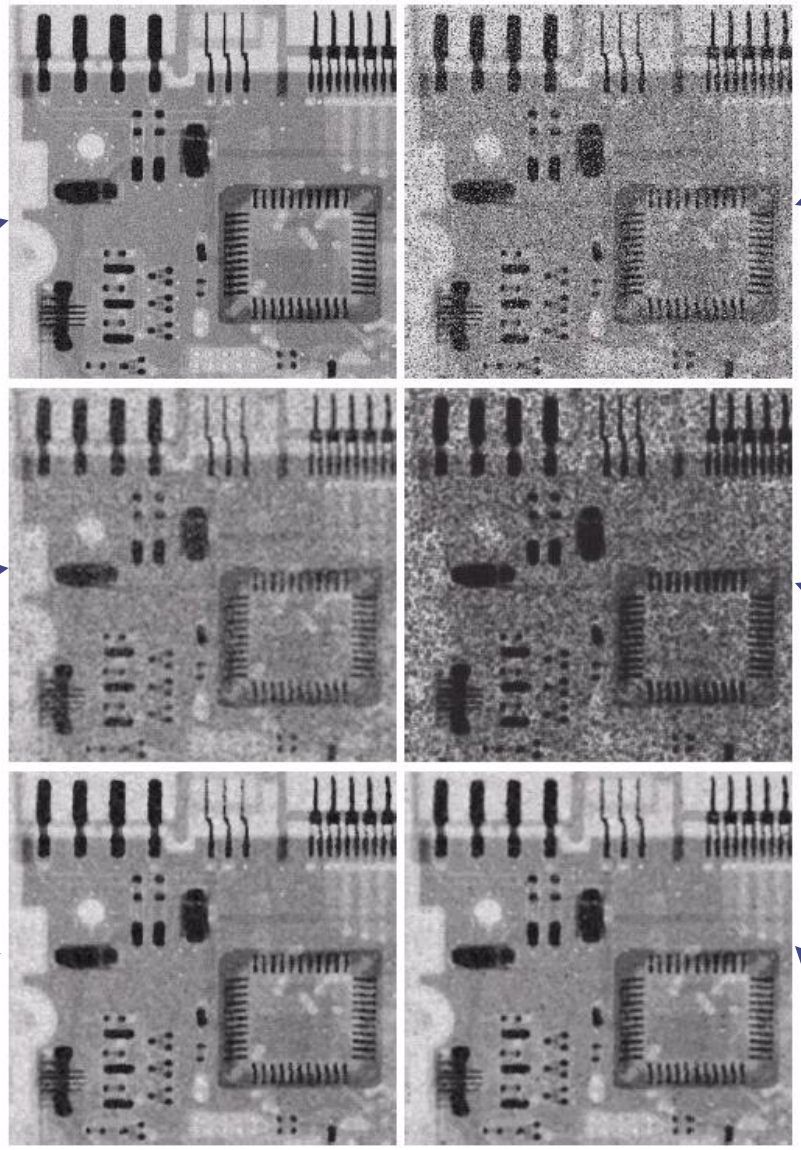
Added salt & pepper noise

5x5 Mean Filtering

5x5 Geo-Mean Filtering

5x5 Median Filtering

5x5 Alpha-trimmed Mean Filtering



a b
c d
e f

FIGURE 5.12 (a) Image corrupted by additive uniform noise. (b) Image additionally corrupted by additive salt-and-pepper noise. Image in (b) filtered with a 5×5: (c) arithmetic mean filter; (d) geometric mean filter; (e) median filter; and (f) alpha-trimmed mean filter with $d = 5$.

Adaptive Filter

- Filters whose behavior changes based on statistical characteristics of the image.
- Two adaptive filters are considered:
  (1) Adaptive, local noise reduction filter.
  (2) Adaptive median filter.

Adaptive, Local Noise Reduction Filter

- Two parameters are considered:
  - Mean: measure of average gray level.
  - Variance: measure of average contrast.
- Four measurements:
  (1) noisy image at (x, y): $g(x, y)$
  (2) The variance of noise σ^2_{η}
  (3) The local mean m_L in S_{xy}
  (4) The local variance σ^2_L

Adaptive, Local Noise Reduction Filter

Given the corrupted image $g(x, y)$, find $f(x, y)$.

Conditions:

(a) σ^2_η is zero (Zero-noise case)
- Simply return the value of $g(x, y)$.

(b) If σ^2_L is higher than σ^2_η
- Could be edge and should be preserved.
- Return value close to $g(x, y)$.

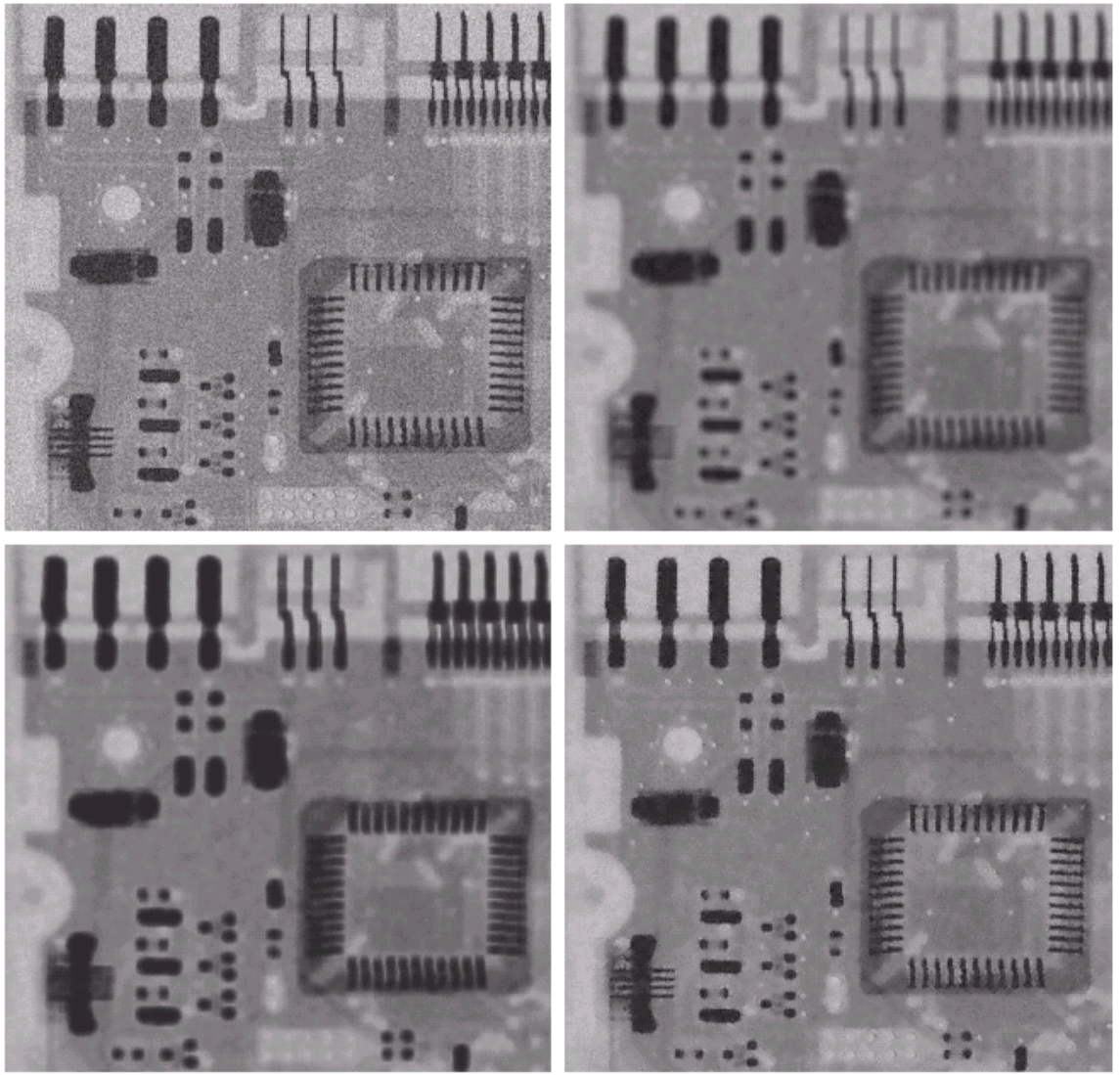
(c) If $\sigma^2_L = \sigma^2_\eta$
- when the local area has similar properties with the overall image.
- Return arithmetic mean value of the pixels in S_{xy}.

General expression: $$\hat{f}(x,y) = g(x,y) - \frac{\sigma^2_\eta}{\sigma^2_L}[g(x,y) - m_L]$$

Adaptive, Local Noise Reduction Filter

a b
c d

FIGURE 5.13
(a) Image corrupted by additive Gaussian noise of zero mean and variance 1000.
(b) Result of arithmetic mean filtering.
(c) Result of geometric mean filtering.
(d) Result of adaptive noise reduction filtering. All filters were of size 7×7.



Adaptive Median Filter (De-Noising)

- Median filter is effective for removing salt-and-pepper noise
  - The density of the impulse noise can not be too large
- Adaptive median filter
  - Notation
    - Z_{min}: minimum gray value in S_{xy}
    - Z_{max}: maximum gray value in S_{xy}
    - Z_{med}: median of gray levels in S_{xy}
    - Z_{xy}: gray value of the image at (x,y)
    - S_{max}: maximum allowed size of S_{xy}

Adaptive Median Filter (De-Noising)

- Two levels of operations
  - Level A:
    - A1= $Z_{med} - Z_{min}$
    - A2= $Z_{med} - Z_{max}$
    - If A1 > 0 AND A2 < 0, Go to level B else increase the window size by 2
    - If window size <= S_{max} repeat level A else output Z_{xy}
  - Level B:
    - B1= $Z_{xy} - Z_{min}$
    - B2= $Z_{xy} - Z_{max}$
    - If B1 > 0 AND B2 < 0, output Z_{xy} else output Z_{med}

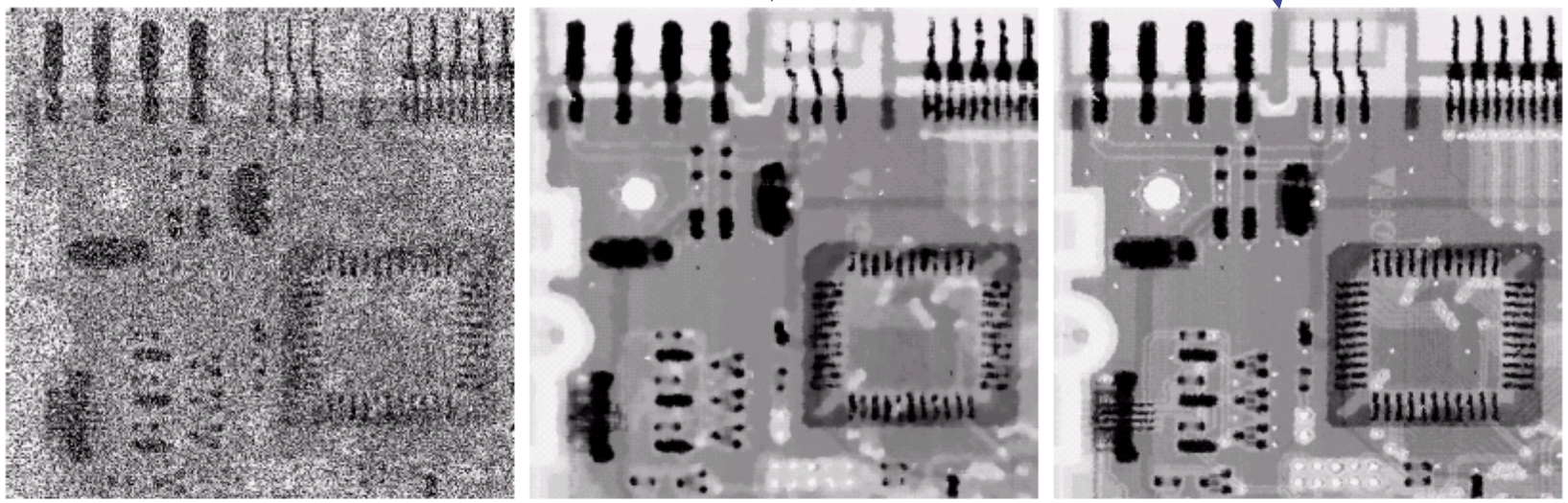
Used to test whether Z_{med} is part of s-and-p noise. If yes, window size is increased

Used to test whether Z_{xy} is part of s-and-p noise. If yes, apply regular median filtering

De-Noising

Median Filtering

Adaptive Median Filtering



a b c

FIGURE 5.14 (a) Image corrupted by salt-and-pepper noise with probabilities $P_a = P_b = 0.25$. (b) Result of filtering with a 7×7 median filter. (c) Result of adaptive median filtering with $S_{max} = 7$.

Periodic Noise Reduction by Frequency Domain Filtering

- **Lowpass and highpass filters for image enhancement have been studied**
- **Bandreject, bandpass, and notch filters as tools for periodic noise reduction or removal are to be studied in this section.**

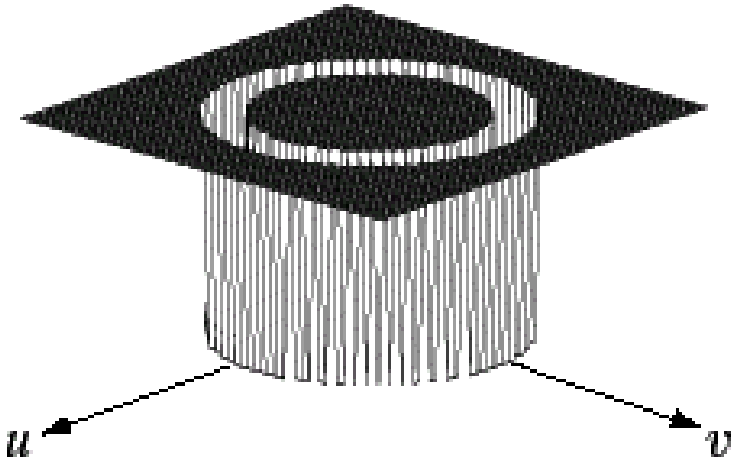
Bandreject Filters

- Bandreject filters remove or attenuate a band of frequencies about the origin of the Fourier transform.
- Similar to those LPFs and HPFs studied, we can construct ideal, Butterworth, and Gaussian bandreject filters

Bandreject Filters

- **Ideal** bandreject filter

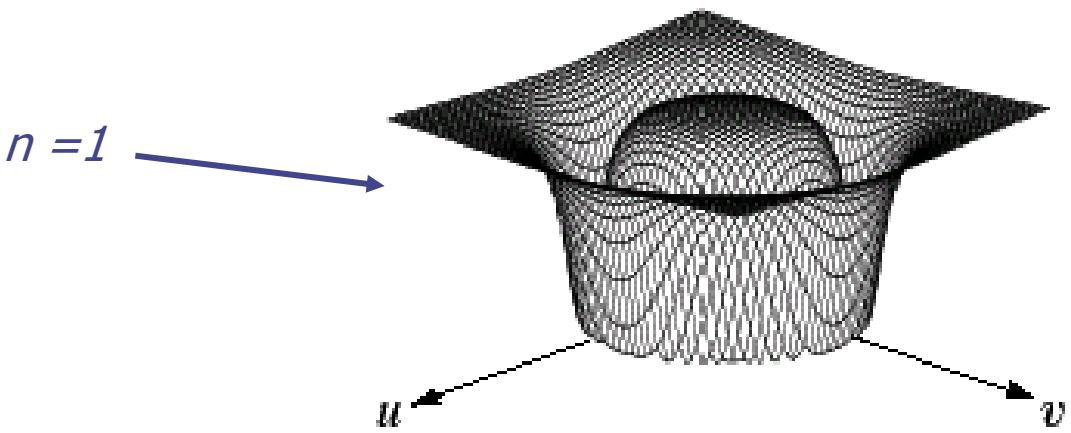
$$H(u,v)=\begin{cases}1 & \text{if } D(u,v)<D_0-\frac{W}{2}\\ 0 & \text{if } D_0-\frac{W}{2}\le D(u,v)\le D_0+\frac{W}{2}\\ 1 & \text{if } D(u,v)>D_0+\frac{W}{2}\end{cases}$$



Bandreject Filters

- **Butterworth** bandreject filter

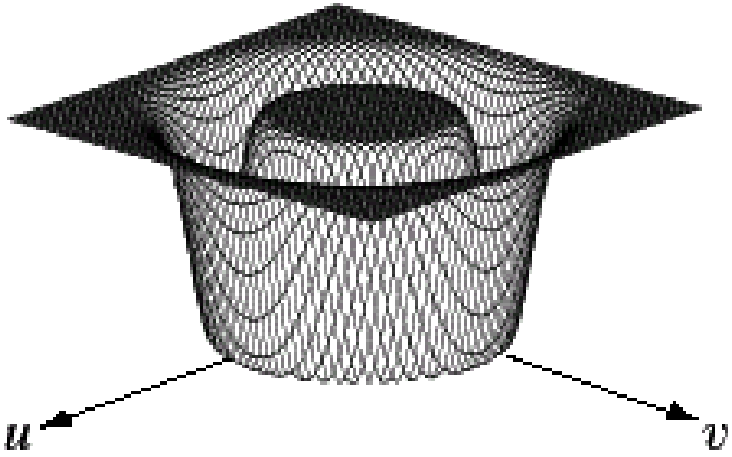
$$H(u,v) = \frac{1}{1 + \left[\frac{D(u,v)W}{D^2(u,v) - D_0^2} \right]^{2n}}$$



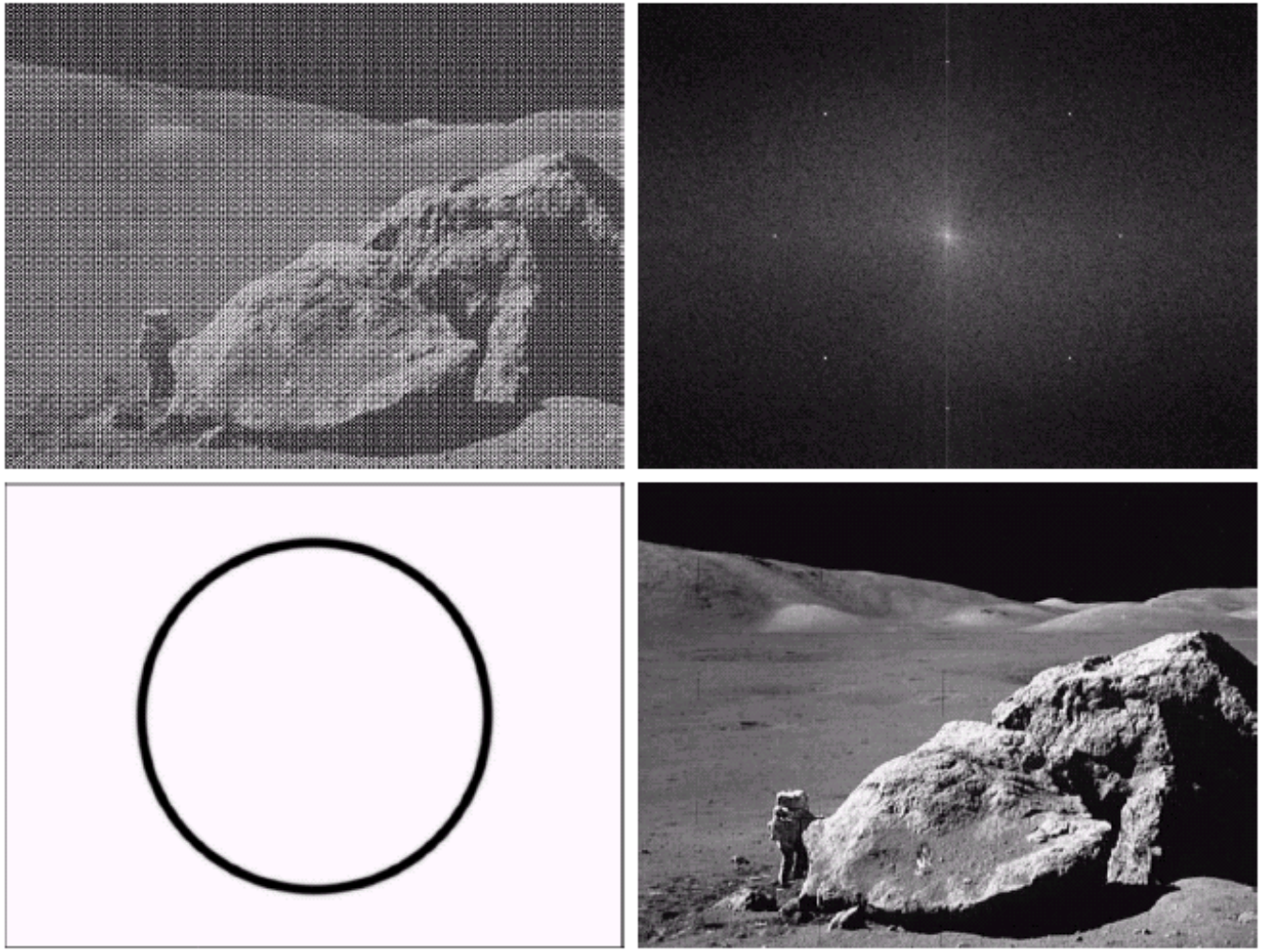
Bandreject Filters

- **Gaussian** bandreject filter

$$H(u,v) = 1 - e^{-\frac{1}{2}\left[\frac{D^2(u,v) - D_0^2}{D(u,v)W}\right]}$$



Bandreject Filters



a b
c d

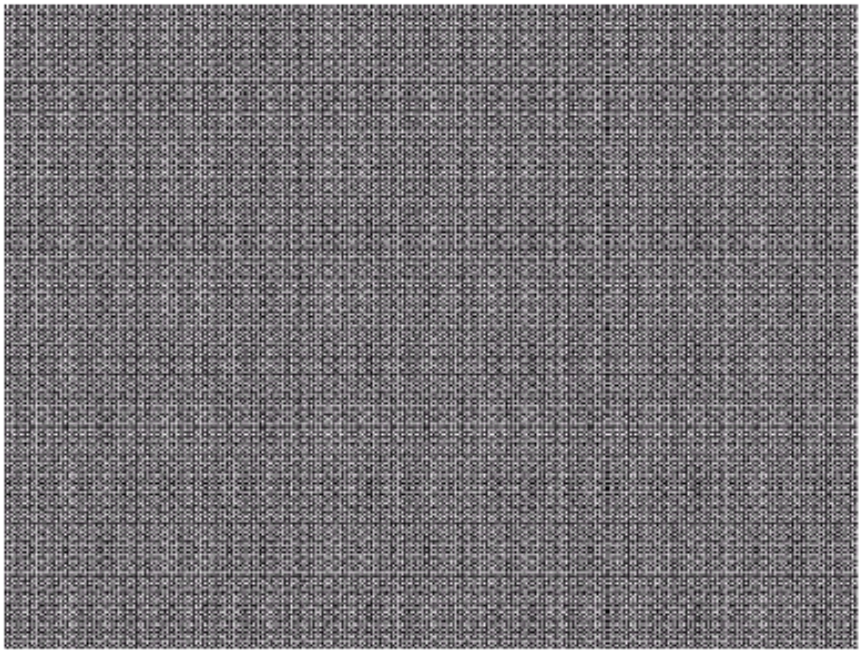
FIGURE 5.16
(a) Image corrupted by sinusoidal noise. (b) Spectrum of (a). (c) Butterworth bandreject filter (white represents 1). (d) Result of filtering. (Original image courtesy of NASA.)

Bandbass Filters

Bandpass filter performs the opposite of a bandreject filter

$$H_{bp}(u,v) = 1 - H_{br}(u,v)$$

FIGURE 5.17
Noise pattern of the image in Fig. 5.16(a) obtained by bandpass filtering.



Notch Filters

- Notch filter ejects frequencies in predefined neighborhoods about a center frequency.
- It appears in symmetric pairs about the origin because the Fourier transform of a real valued image is symmetric.

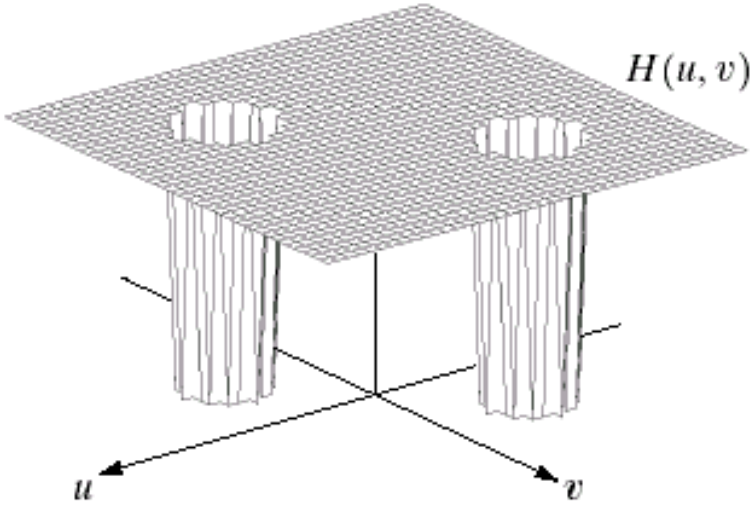
Notch Filters

- **Ideal** notch filter

$$H(u,v)=\begin{cases}0 & \text{if } D_1(u,v)\le D_0 \text{ or } D_2(u,v)\le D_0\\ 1 & \text{otherwise}\end{cases}$$

$$D_1(u,v)=\left[(u-M/2-u_0)^2+(v-N/2-v_0)^2\right]^{1/2}$$

$$D_2(u,v)=\left[(u-M/2+u_0)^2+(v-N/2+v_0)^2\right]^{1/2}$$

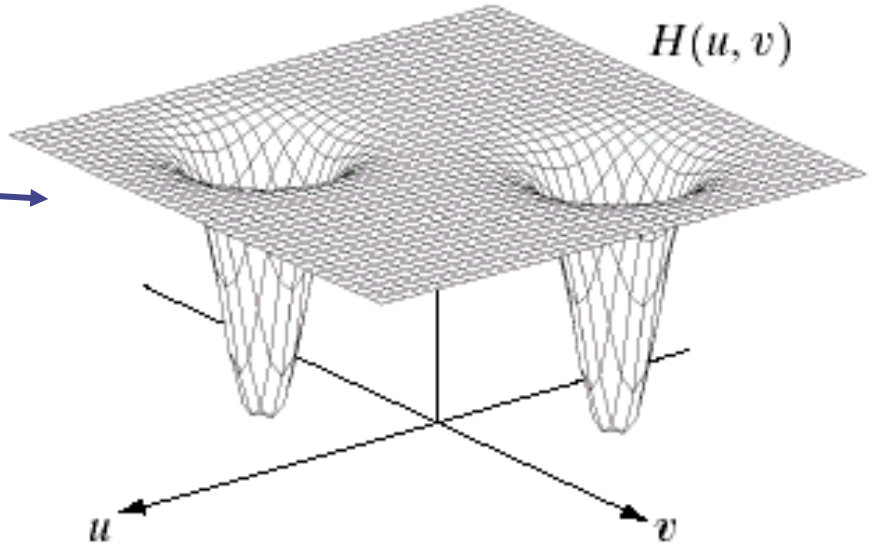


Notch Filters

- **Butterworth** notch filter

$$H(u,v) = \frac{1}{1+\left[\dfrac{D_0^2}{D_1(u,v)D_2(u,v)}\right]^{2n}}$$

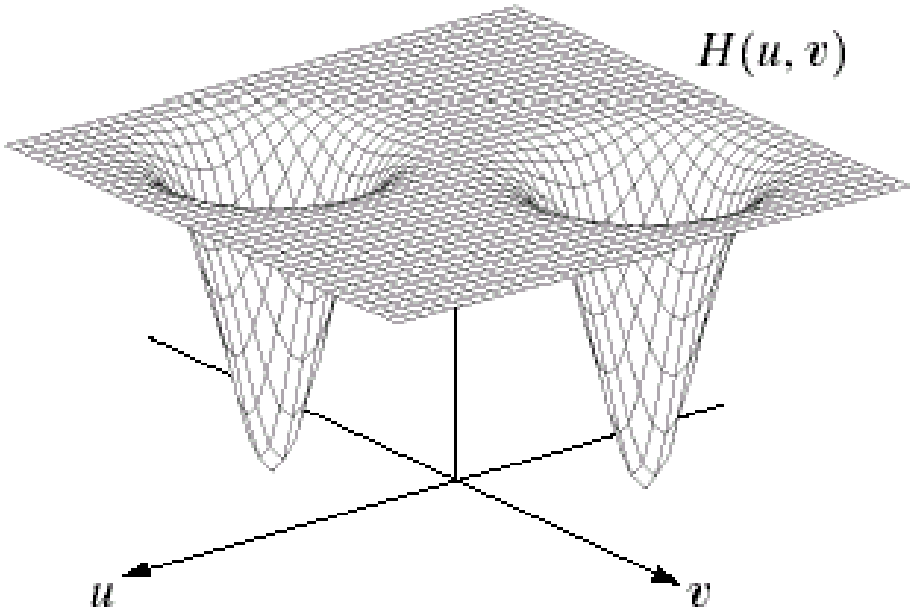
n=2



Notch Filters

- **Gaussian** notch filter

$$H(u,v) = 1 - e^{-\frac{1}{2}\left[\frac{D_1(u,v)D_2(u,v)}{D_0^2}\right]}$$



Notch Filters

Notch filters that pass, rather than suppress:

$$H_{np}(u,v) = 1 - H_{nr}(u,v)$$

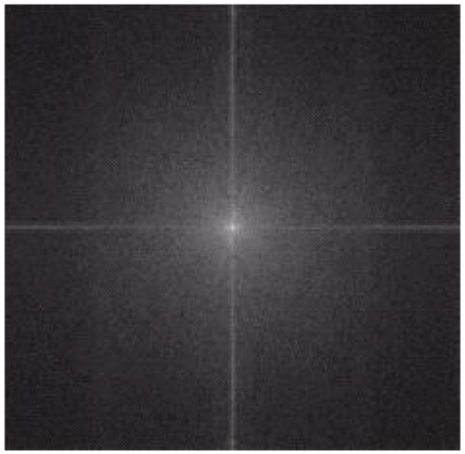
- *NR* filters become highpass filters if $u_0 = v_0 = 0$

- *NP* filters become lowpass filters if $u_0 = v_0 = 0$

Notch Filters

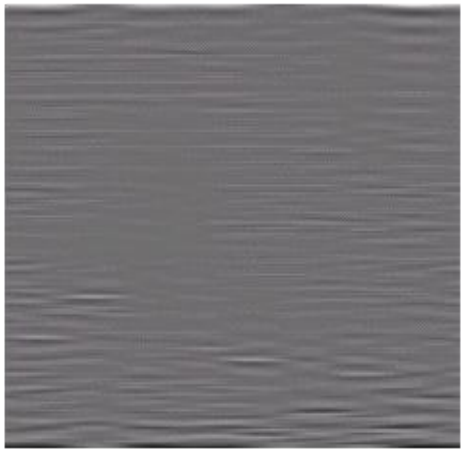
You can see the effect of scan lines

Spectrum of image



Notch pass filter

IFT of NP filtered image

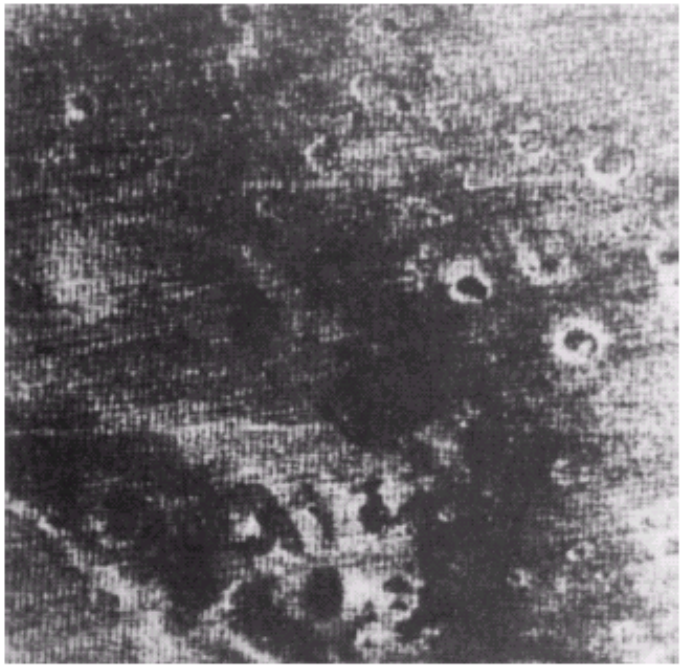


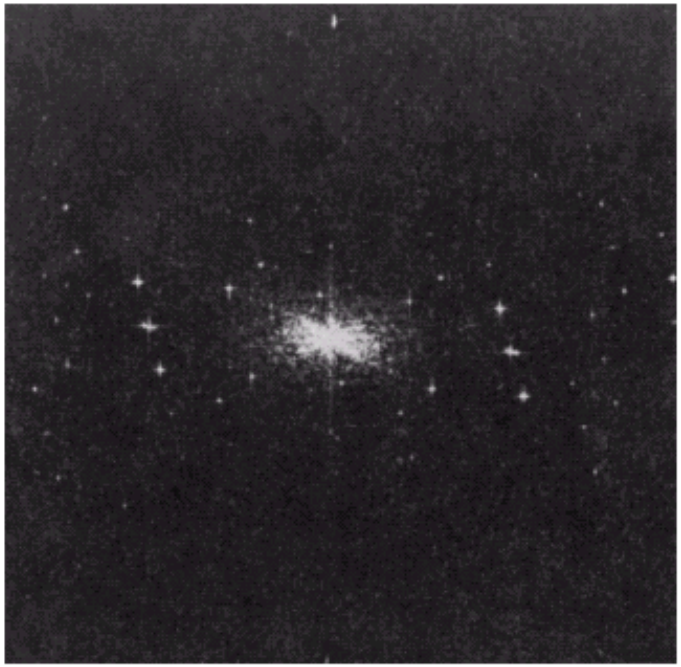
Result of NR filter

Optimum Notch Filtering

a b

FIGURE 5.20
(a) Image of the Martian terrain taken by *Mariner 6.*
(b) Fourier spectrum showing periodic interference.
(Courtesy of NASA.)





Optimum Notch Filtering

- In the ideal case, the original image can be restored if the noise can be estimated completely.
  - That is: $f(x,y) = g(x,y) - \eta(x,y)$

- However, the noise can be only partially estimated. This means the restored image is not exact.
  - Which means $\hat{f}(x,y) = g(x,y) - \hat{\eta}(x,y)$

$$\hat{\eta}(x,y) = IFT\{H(u,v)G(u,v)\}$$

Optimum Notch Filtering

- In this section, we try to improve the restored image by introducing a modulation function
  - $\hat{f}(x,y) = g(x,y) - w(x,y)\hat{\eta}(x,y)$
  - Here the modulation function is a constant within a neighborhood of size (2a+1) by (2b+1) about a point (x,y)
  - We optimize its performance by minimizing the local variance of the restored image at the position (x,y)

$$\sigma^2(x,y) = \frac{1}{(2a+1)(2b+1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} \left[\hat{f}(x+s, y+t) - \bar{\hat{f}}(x,y) \right]^2$$

$$\bar{\hat{f}}(x,y) = \frac{1}{(2a+1)(2b+1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} \hat{f}(x+s, y+t)$$

Optimum Notch Filtering

Points on or near Edge of the image can be treated by considering partial neighborhoods

$$\sigma^2(x,y) = \frac{1}{(2a+1)(2b+1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} \{[g(x+s,y+t) - w(x+s,y+t)\hat{\eta}(x+s,y+t)] - [\bar{g}(x,y) - \overline{w(x,y)\hat{\eta}(x,y)}]\}^2$$

Assumption: $w(x+s,y+t) = w(x,y) \quad for \quad -a \le s \le a \ and \ -b \le t \le b$

$$\Rightarrow \quad \overline{w(x,y)\hat{\eta}(x,y)} = w(x,y)\bar{\hat{\eta}}(x,y)$$

Optimum Notch Filtering

$$\sigma^2(x,y) = \frac{1}{(2a+1)(2b+1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} \{[g(x+s,y+t) - w(x+s,y+t)\hat{\eta}(x+s,y+t)] - [\bar{g}(x,y) - w(x,y)\bar{\hat{\eta}}(x,y)\}^2$$

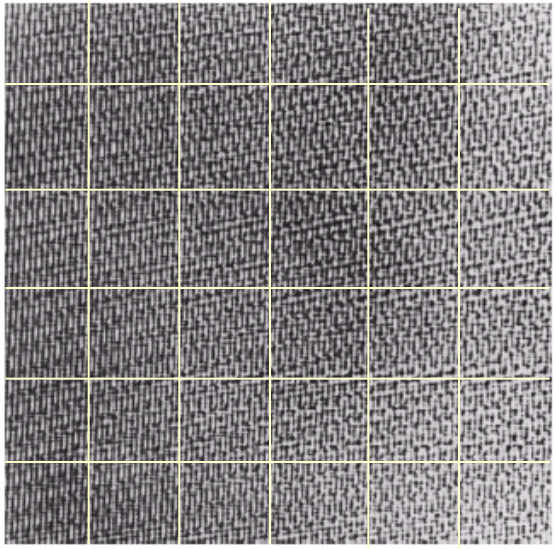
To minimize $\sigma^2(x,y)$

$$\frac{\partial \sigma^2(x,y)}{\partial w(x,y)} = 0$$

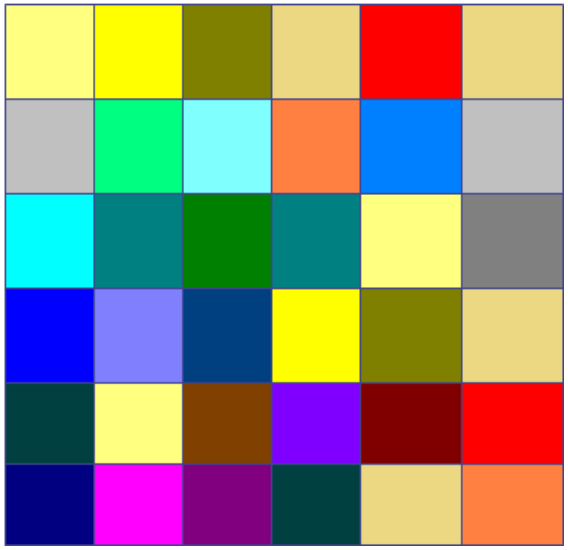
$$\Rightarrow w(x,y) = \frac{\overline{g(x,y)\hat{\eta}(x,y)} - \bar{g}(x,y)\bar{\hat{\eta}}(x,y)}{\overline{\hat{\eta}^2}(x,y) - \bar{\hat{\eta}}^2(x,y)}$$

Optimum Notch Filtering

$g(x,y)$



$\hat{\eta}(x,y)$



$w(x,y)$

$$\hat{f}(x,y) = g(x,y) - w(x,y)\hat{\eta}(x,y)$$

Optimum Notch Filtering

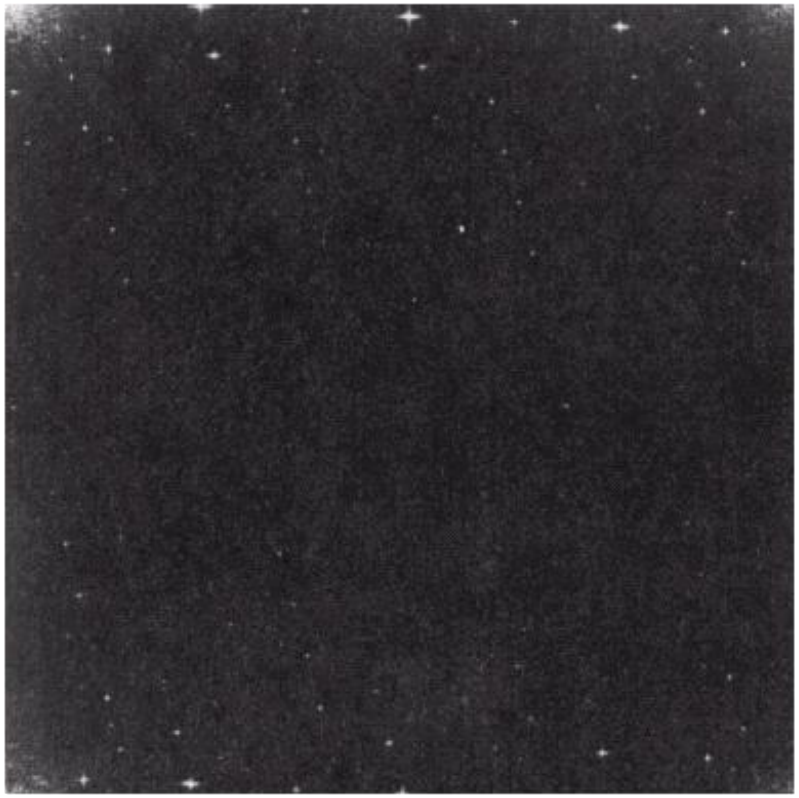
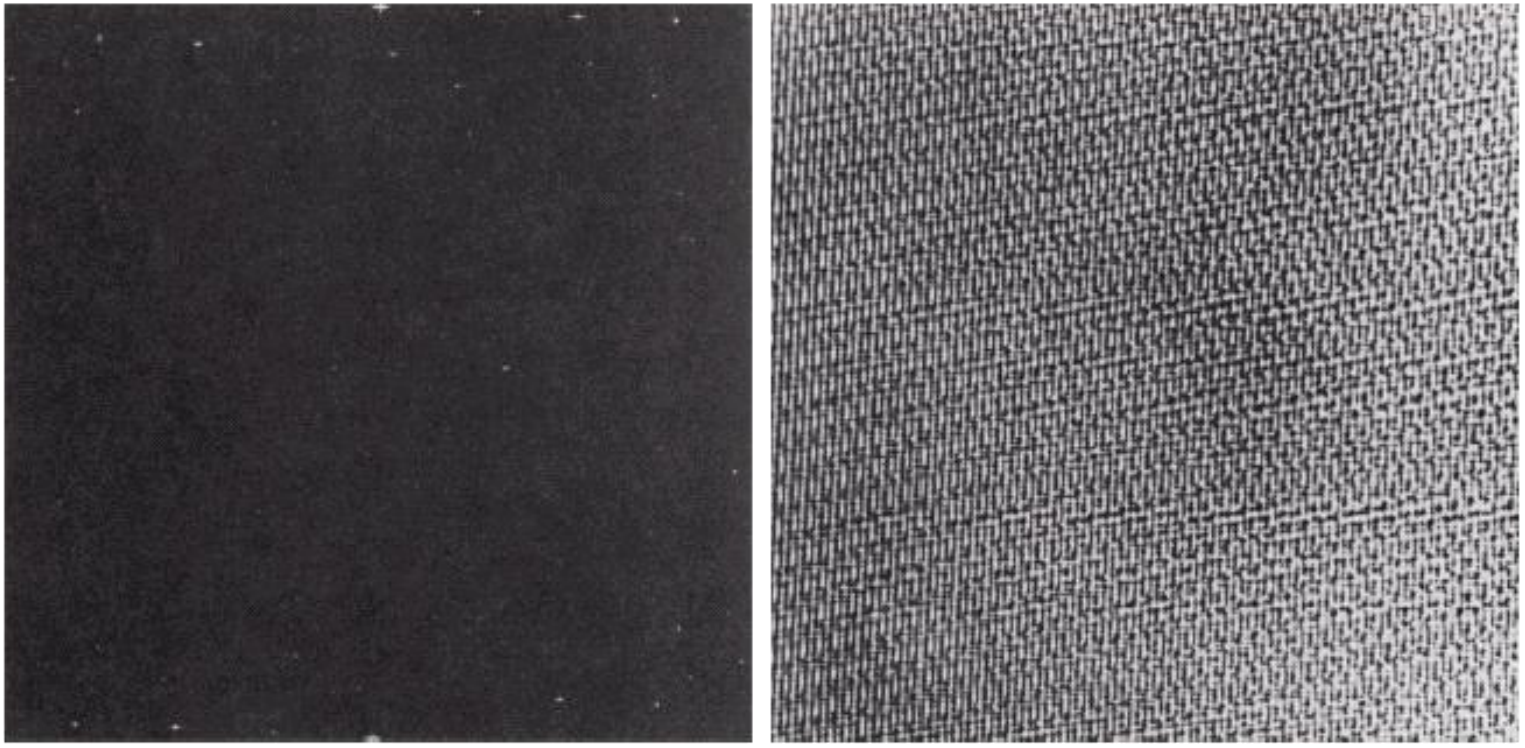


FIGURE 5.21 Fourier spectrum (without shifting) of the image shown in Fig. 5.20(a). (Courtesy of NASA.)

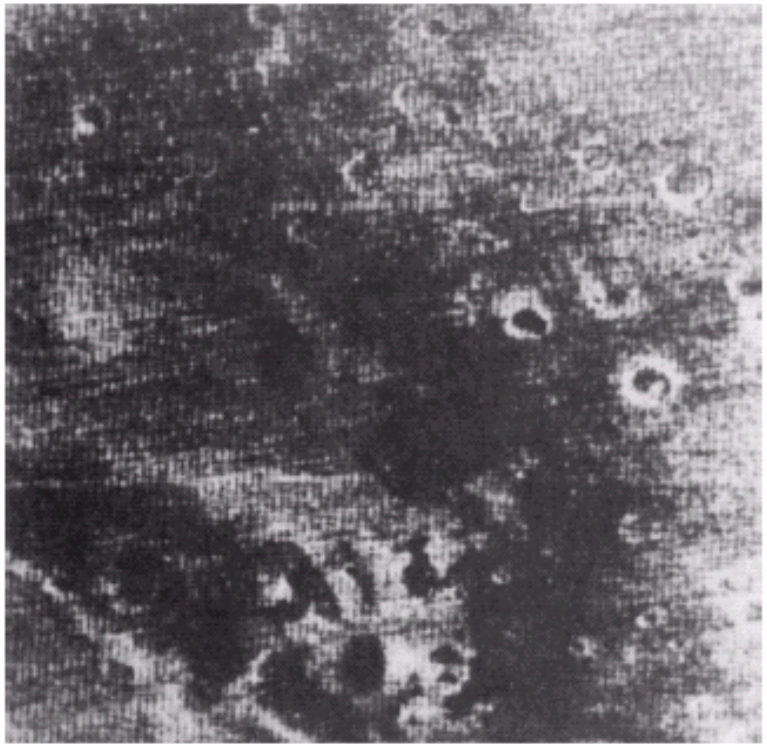
Optimum Notch Filtering



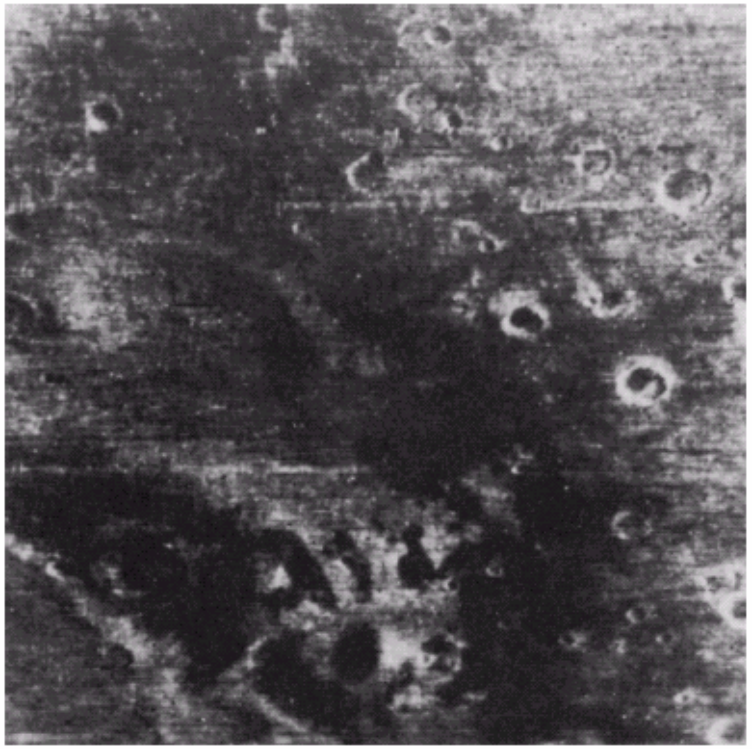
a b

FIGURE 5.22 (a) Fourier spectrum of $N(u, v)$, and (b) corresponding noise interference pattern $\eta(x, y)$. (Courtesy of NASA.)

Optimum Notch Filtering



$g(x,y)$



$\hat{f}(x,y)$

Image size: 512x512

a=b=15

Linear, Position-Invariant Degradation

- Degradation Model

$$g(x,y) = H[\,f(x,y)] + \eta(x,y)$$

In the absence of additive noise:

For scalar values of a and b, H is linear if:

$$H[a\ f_1(x,y) + b\ f_2(x,y)] = a\,H[\,f_1(x,y)] + b\,H[\,f_2(x,y)]$$

H is Position-Invariant if:

$$g(x,y) = H[\,f(x,y)] \Rightarrow H[\,f(x-\alpha, y-\beta)] = g(x-\alpha, y-\beta)$$

Linear, Position-Invariant Degradation

In the presence of additive noise:

$$g(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(\alpha,\beta)h(x-\alpha,y-\beta)d\alpha\, d\beta + \eta(x,y)$$

$$g(x,y) = h(x,y) * f(x,y) + \eta(x,y)$$

$$G(u,v) = H(u,v)F(u,v) + N(u,v)$$

- **Many types of degradation can be approximated by linear, position-invariant processes**
- **Extensive tools of linear system theory are available**
- **In this situation, restoration is *image deconvolution***

Estimating the Degradation Function

- **Principal way to estimate the degradation function for use in image restoration:**
  - **Observation**
  - **Experimentation**
  - **Mathematical modeling**

Estimating by Image Observation

- **We look for a small section of the image that has strong signal content ($g_s(x,y)$) and then construct an un-degradation of this section by using sample gray levels ($\hat{f}_s(x,y)$).**

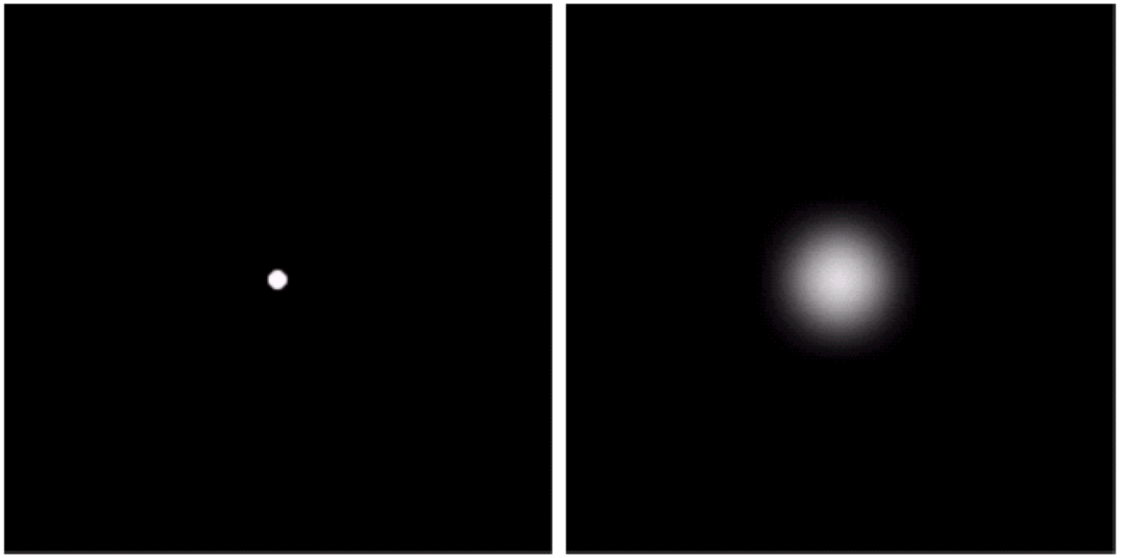
$$H_s(u,v) = \frac{G_s(u,v)}{\hat{F}_s(u,v)}$$

Now, we construct a function $H(u,v)$ on a large scale, but having the same shape.

Estimating by Experimentation

- We try to obtain impulse response of the degradation by imaging an impulse (small dot of light) using the system. Therefore

$$H(u,v) = \frac{G(u,v)}{A}$$



a b

FIGURE 5.24 Degradation estimation by impulse characterization. (a) An impulse of light (shown magnified). (b) Imaged (degraded) impulse.

Estimating by Modeling

Atmospheric turbulence model: $H(u,v) = e^{-k(u^2+v^2)^{5/6}}$

Negligible turbulence

High turbulence k=0.0025

Mid turbulence k=0.001

Low turbulence k=0.00025

Estimating by Modeling

Blurring by linear motion:

$$g(x,y) = \int_0^T f[x - x_0(t), y - y_0(t)]dt$$

$$G(u,v) = F(u,v)\int_0^T e^{-j2\pi[ux_0(t)+vy_0(t)]}dt$$

$$\Rightarrow H(u,v) = \int_0^T e^{-j2\pi[ux_0(t)+vy_0(t)]}dt$$

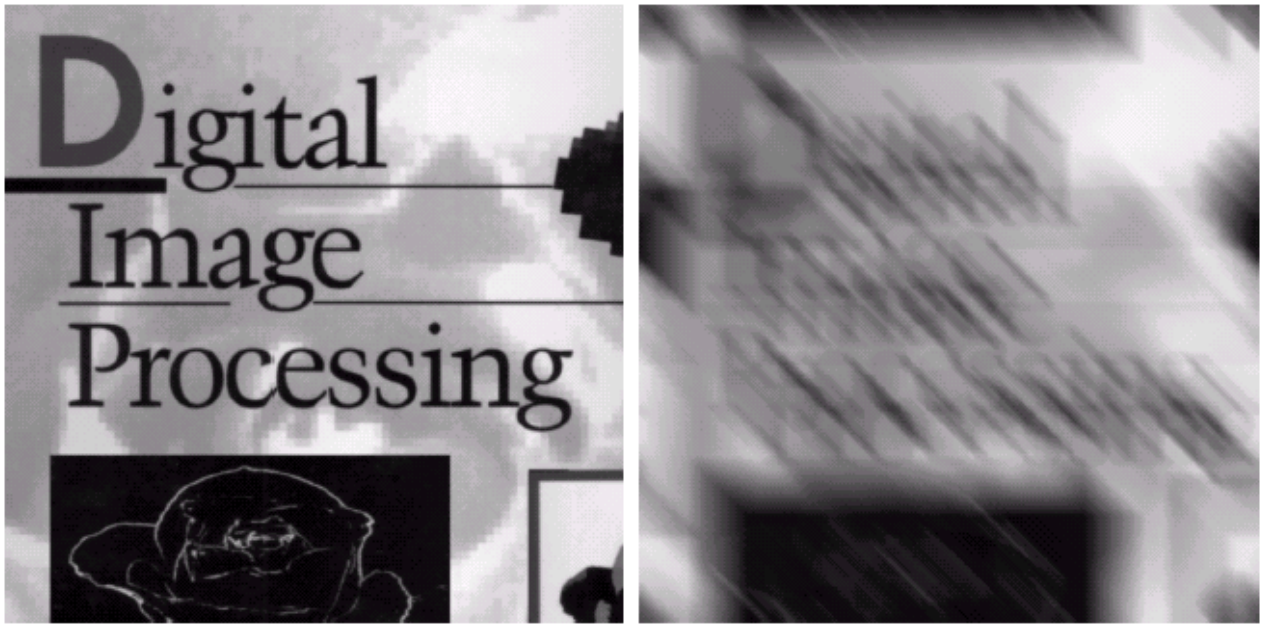
$$if \;\; x_0(t) = at/T \;\; and \;\; y_0(t) = 0 \Rightarrow H(u,v) = \int_0^T e^{-2\pi uat/T}dt$$

$$= \frac{T}{\pi ua}\sin(\pi ua)e^{-j\pi ua}$$

Estimating by Modeling

$$\text{if } x_0(t) = at/T \text{ and } y_0(t) = bt/T \Rightarrow$$

$$H(u,v) = \frac{T}{\pi(ua+vb)}\sin[\pi(ua+vb)]e^{-j\pi(ua+vb)}$$



a b

FIGURE 5.26 (a) Original image. (b) Result of blurring using the function in Eq. (5.6-11) with $a = b = 0.1$ and $T = 1$.

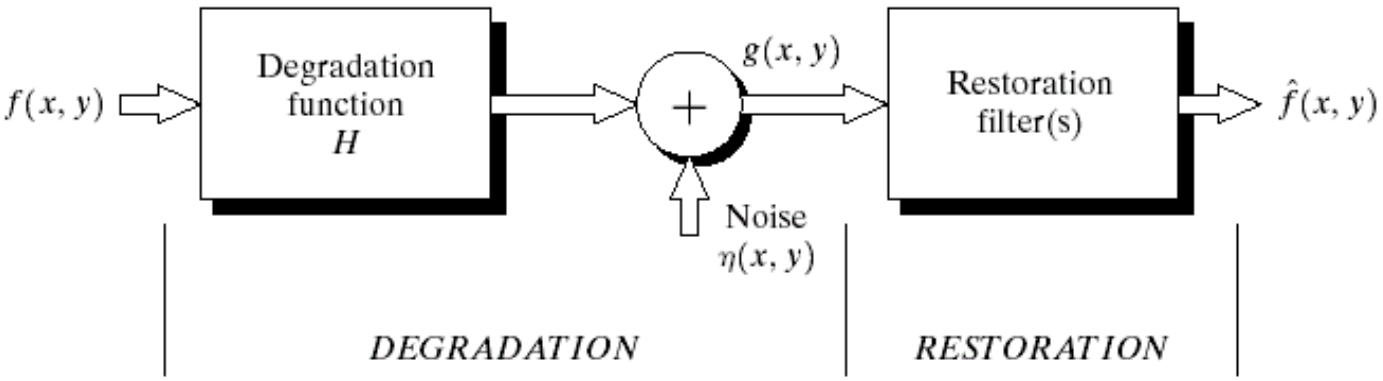
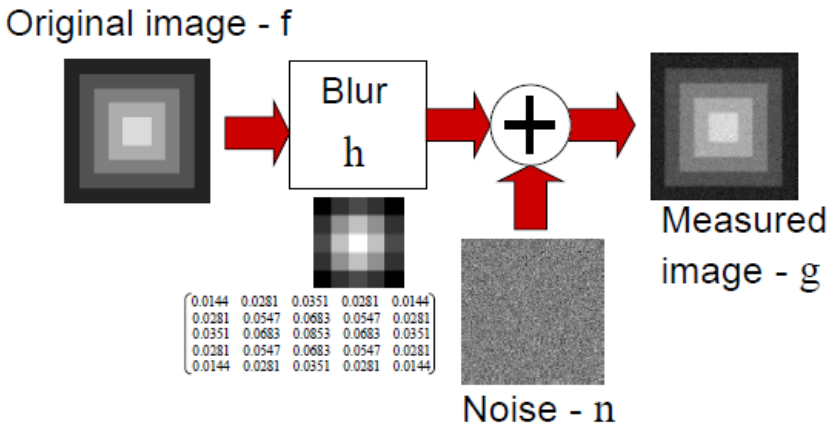


FIGURE 5.1 A model of the image degradation/restoration process.

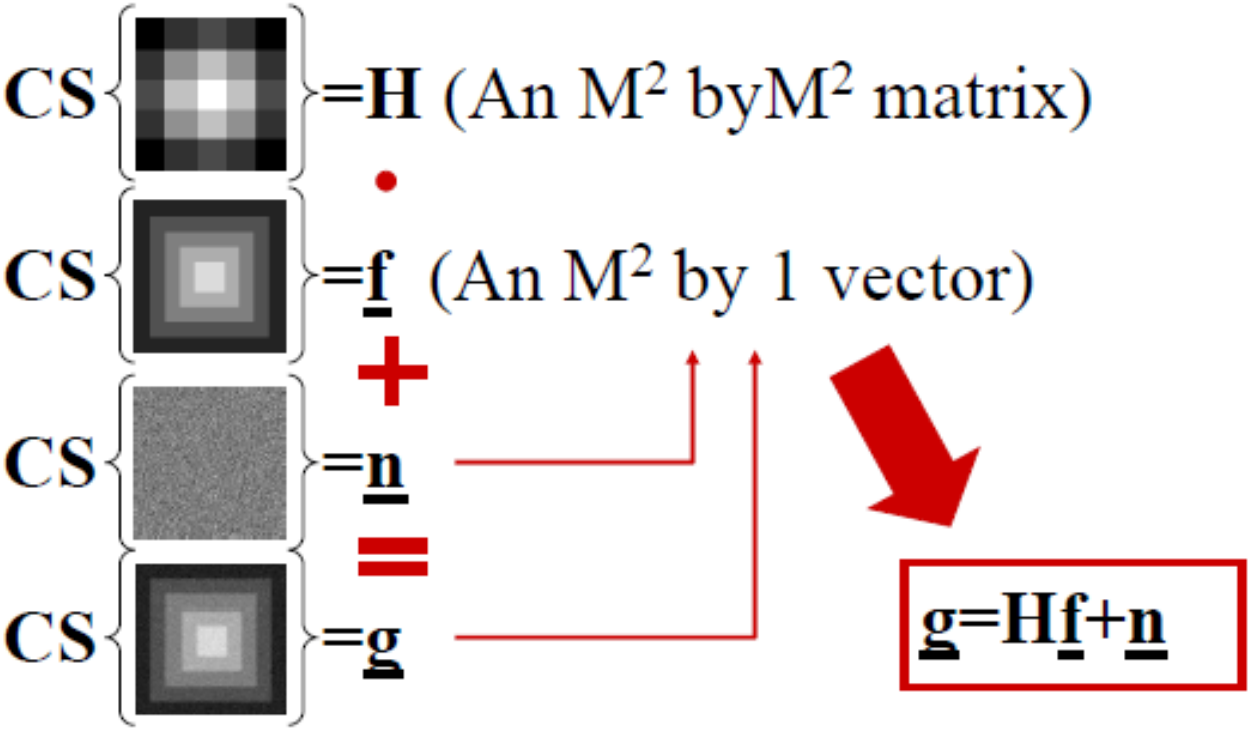
The Forward Model



f,g,n **of size** $M \times M$

Slides credits to Milanfar Peyman

The Forward Model: Matrix Formulation



Matrix formulation of general linear processing:

$$f(x,y)=\begin{bmatrix} f(0,0) & \cdots & f(0,L-1) \\ \vdots & \ddots & \vdots \\ f(N-1,0) & \cdots & f(N-1,L-1) \end{bmatrix}$$

$$\vec{f}=\begin{pmatrix} f(0,0) \\ f(1,0) \\ \vdots \\ f(N-1,0) \\ f(0,1) \\ \vdots \\ f(N-1,1) \\ \vdots \\ \vdots \\ f(0,L-1) \\ \vdots \\ f(N-1,L-1) \end{pmatrix}=\begin{pmatrix} f_{00} \\ f_{01} \\ \vdots \\ f_{0,N-1} \\ f_{10} \\ \vdots \\ f_{N-1,1} \\ \vdots \\ \vdots \\ f_{L-1,0} \\ \vdots \\ f_{L-1,N-1} \end{pmatrix}$$

Column vector of length LxN.

This often makes the mathematics easier.

Column-stacking (CS) operation:

$$\mathbf{CS}\left\{\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}\right\} = \begin{pmatrix} 1 \\ 4 \\ 7 \\ 2 \\ 5 \\ 8 \\ 3 \\ 6 \\ 9 \end{pmatrix}$$

Using the CS ordering, a linear operator on an image can be represented as a matrix multiplying a vector

Column-stacking (CS) operation:

Image of size $N \times N$: a linear operation is represented as a matrix of size $N^2 \times N^2$

$$\underline{g} = \mathbf{CS}\{f * h\} = \mathbf{H} \cdot \mathbf{CS}\{f\} = \mathbf{H}\underline{f}$$

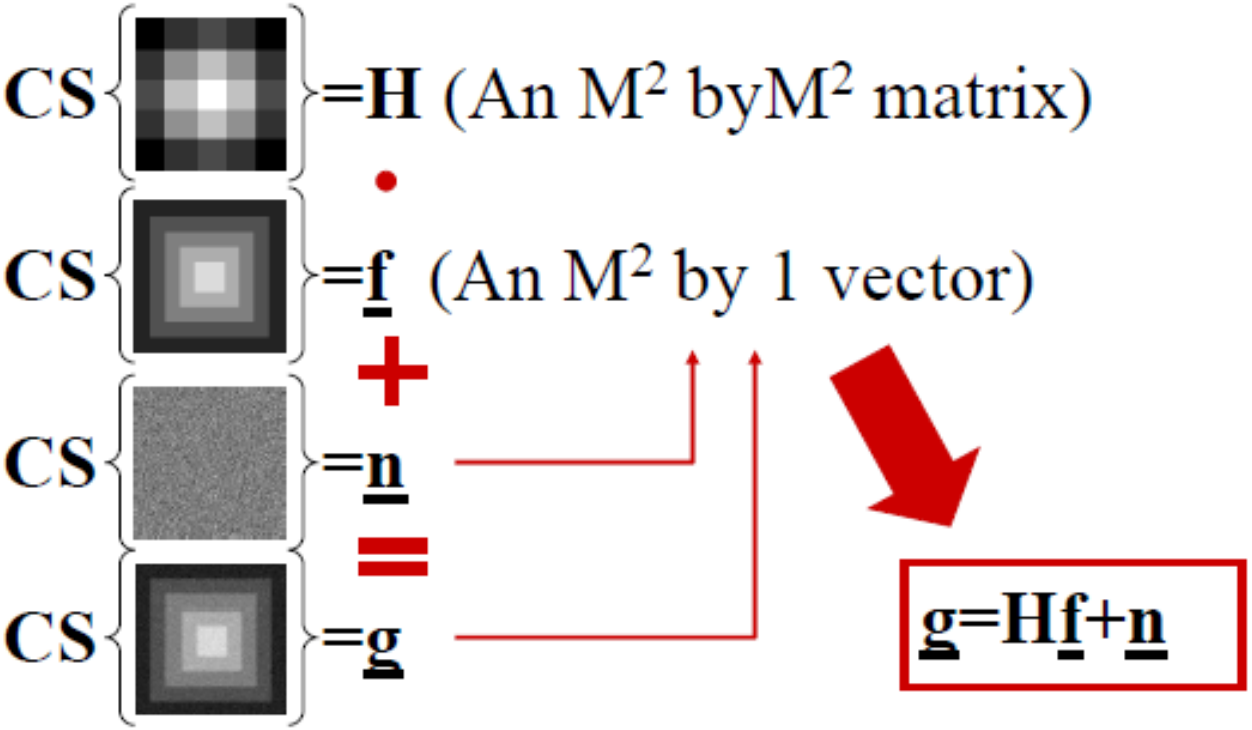
$$\mathbf{CS}\left\{ \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \right\} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \\ 7 \\ 2 \\ 5 \\ 8 \\ 3 \\ 6 \\ 9 \end{bmatrix}$$

The Block-Circulant form:

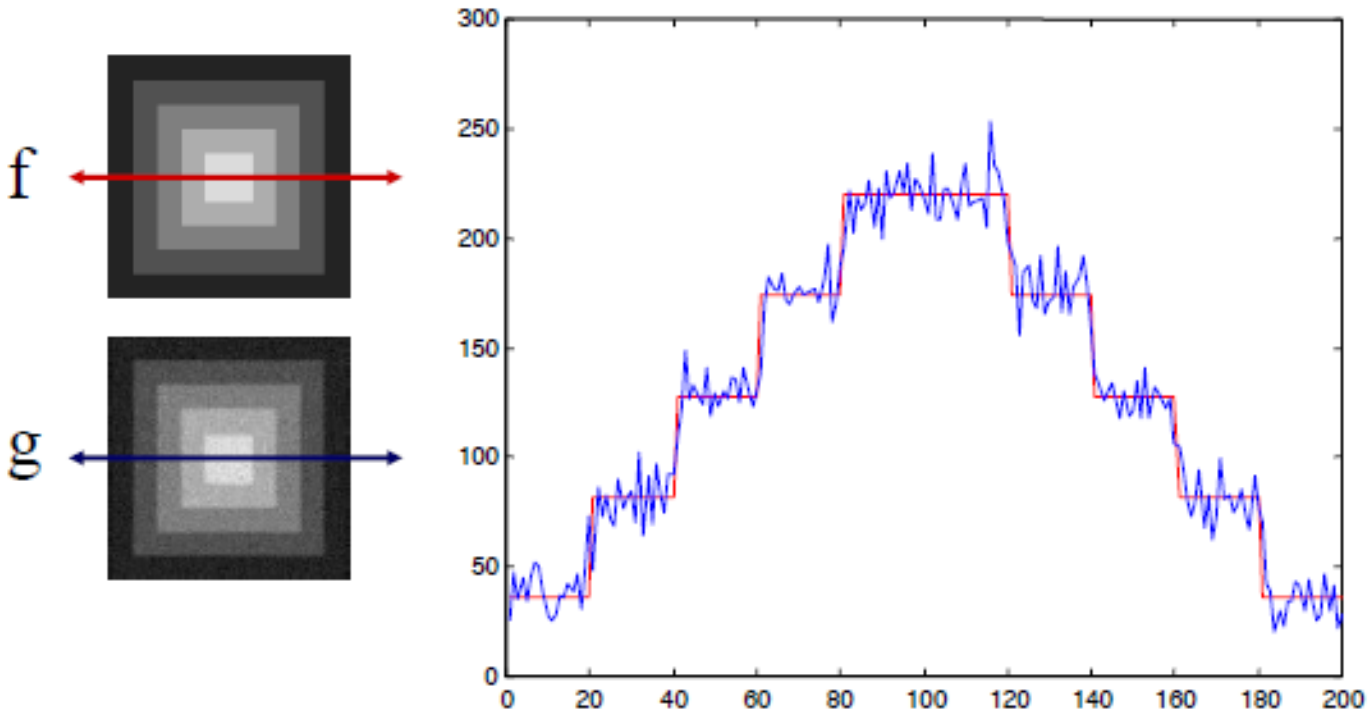
If **H** represents a space invariant operation, it has a block circulant form

$$\begin{pmatrix} A & B & C & D & E & F & G \\ G & A & B & C & D & E & F \\ F & G & A & B & C & D & E \\ E & F & G & A & B & C & D \\ D & E & F & G & A & B & C \\ C & D & E & F & G & A & B \\ B & C & D & E & F & G & A \end{pmatrix}$$

The Forward Model: Matrix Formulation



A section through the image



Sources of Noise

Three major sources:

- During acquisition (often random)
- During transmission
- Effects of coding/decoding

Image noise as a random variable:

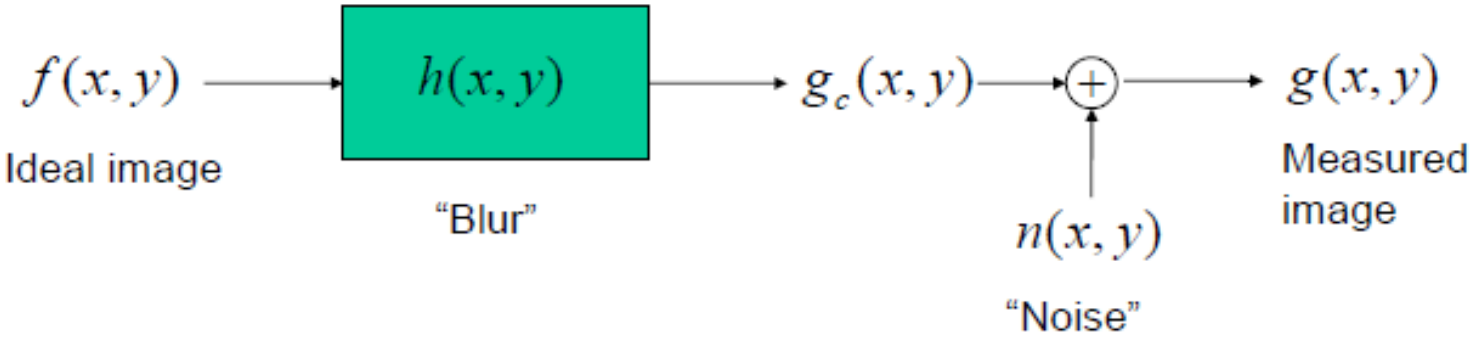
- For every (x,y), n(x,y) can be considered as a random variable.

- In general, we assume that the noise n(x,y) is not dependent on the underlying signal f(x,y).

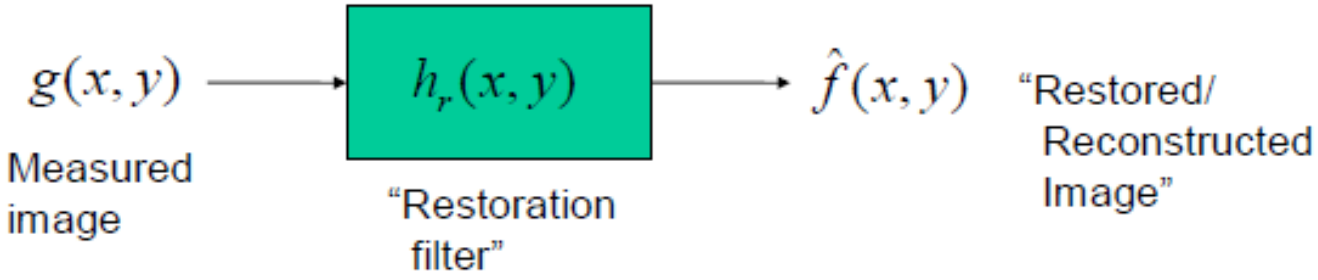
- In general, we assume that the value of n(x,y) is not correlated with n(x',y'). (Spatially uncorrelated noise)

Linear Image Restoration:

PROBLEM:

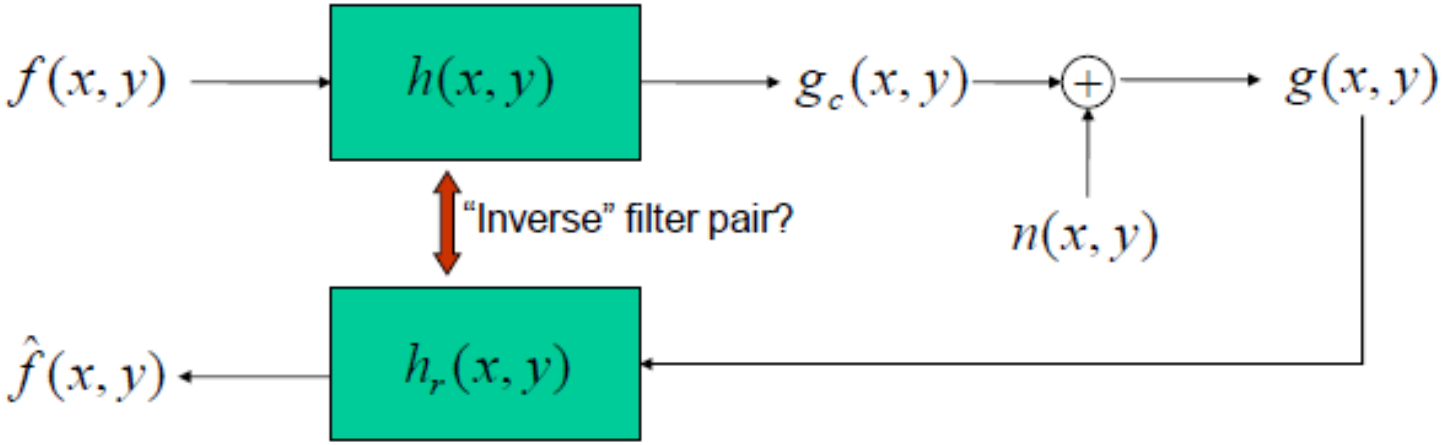


A SOLUTION:



$$\hat{f}(x,y) = g(x,y) * h_r(x,y)$$

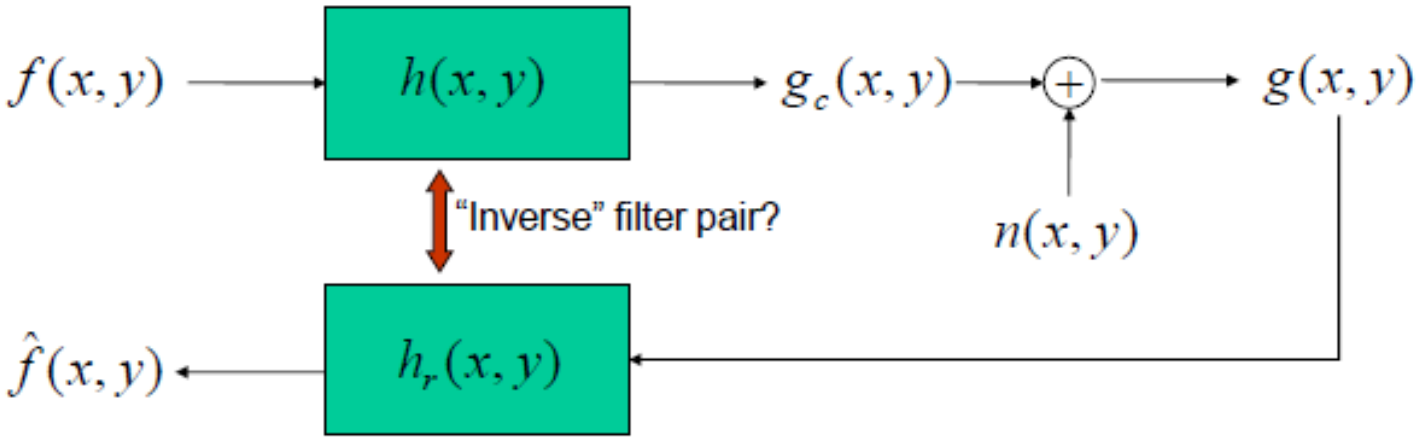
Image Restoration Is a Hard Problem:



Often $h(x,y)$ is a low-pass type filter. (e.g. Motion blur)

$$\hat{f}(x,y) = g(x,y) * h_r(x,y) = \big(f(x,y) * h(x,y) + n(x,y)\big) * h_r(x,y)$$
$$= f(x,y) * h(x,y) * h_r(x,y) + \underbrace{n(x,y) * h_r(x,y)}_{\text{NOISE AMPLIFICATION}}$$

Image Restoration Is a Hard Problem:



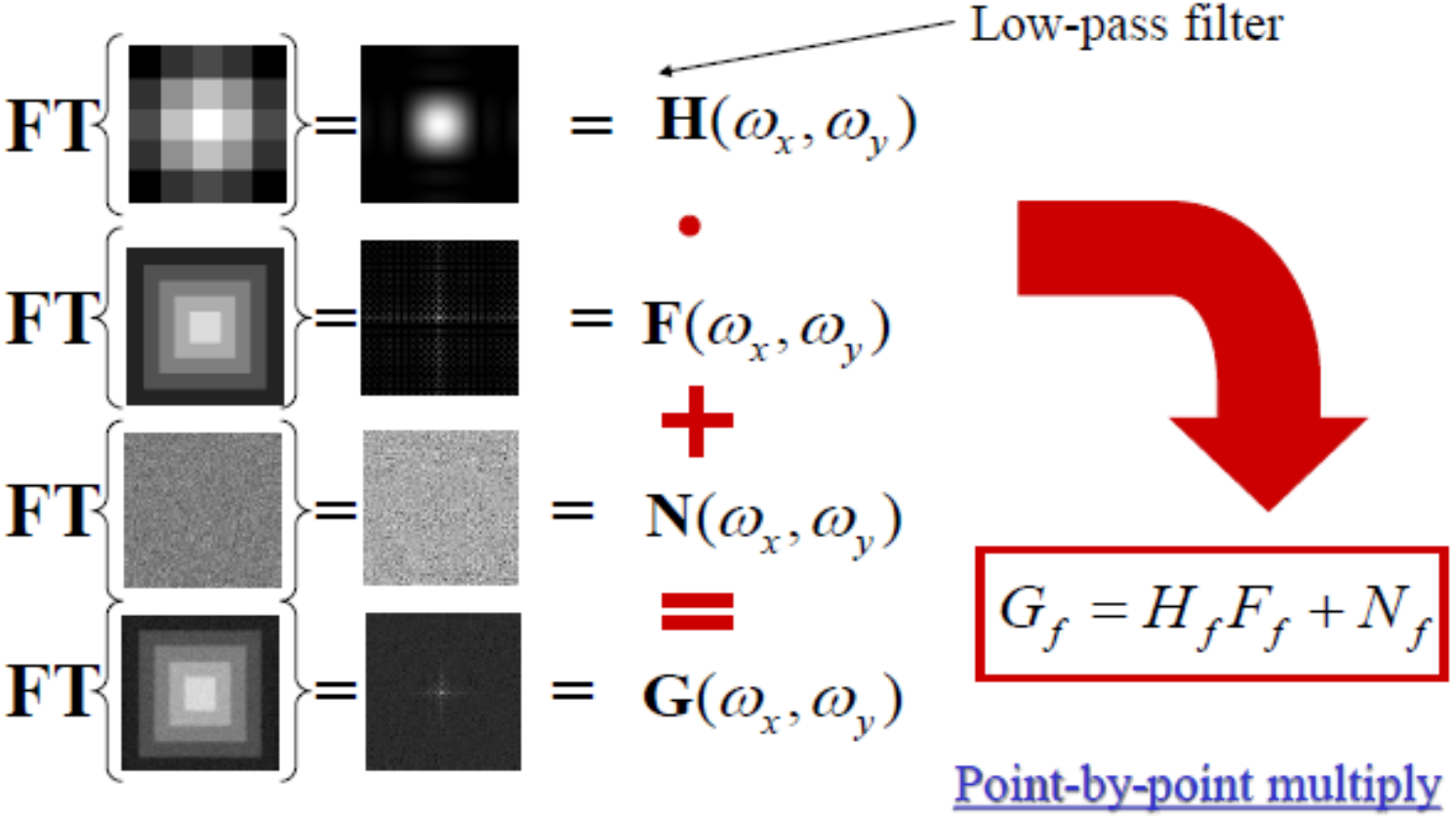
Often $h(x,y)$ is a low-pass type filter. (e.g. Motion blur)

$$\hat{f}(x,y) = g(x,y) * h_r(x,y) = \big(f(x,y) * h(x,y) + n(x,y)\big) * h_r(x,y)$$
$$= f(x,y) * h(x,y) * h_r(x,y) + \underbrace{n(x,y) * h_r(x,y)}$$

NOISE AMPLIFICATION

Image Restoration: Basics of the Frequency Domain Approaches

Frequency Domain Formulation:



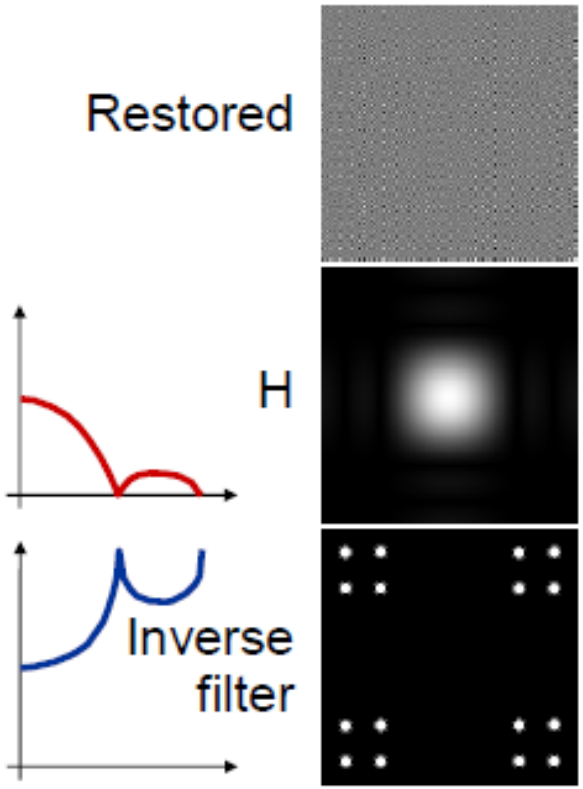
Basic Inversion Idea

$$G_f = H_f F_f + N_f$$

Neglect the noise

$$H_f^{-1} G_f = F_f + H_f^{-1} N_f$$

Element-by-Element inverse



Inverse Filtering

The simplest approach to restoration is direct inverse filtering:

$$\hat{F}(u,v) = \frac{G(u,v)}{H(u,v)}$$

$$\hat{F}(u,v) = F(u,v) + \frac{N(u,v)}{H(u,v)}$$

Even if we know the degradation function, we cannot recover the un-degraded image

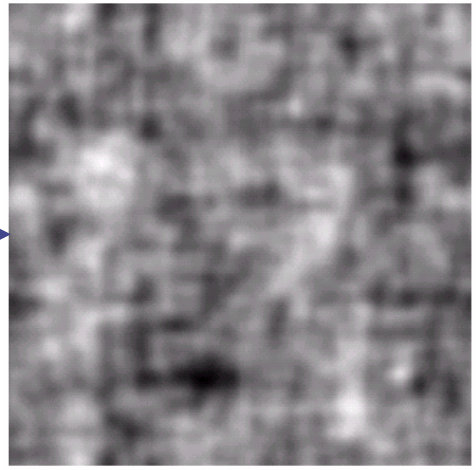
If the degradation has zero or very small values, then the ratio *N/H could easily dominate our estimation of F* .

One approach to get around the zero or small-value problem is to limit the filter frequencies to value near the origin.

Inverse Filtering

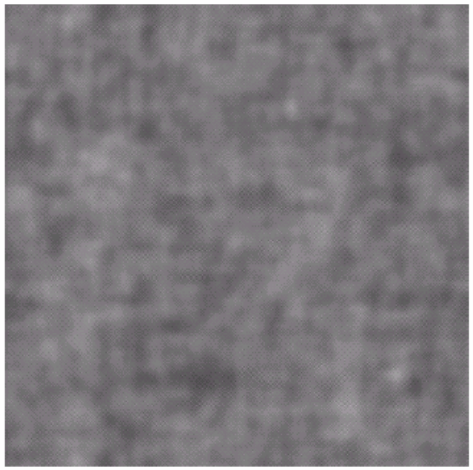
Degraded Image

Full inverse Filtering



Filtering with H cut off outside a radius of 40

Filtering with H cut off outside a radius of 70



Filtering with H cut off outside a radius of 85

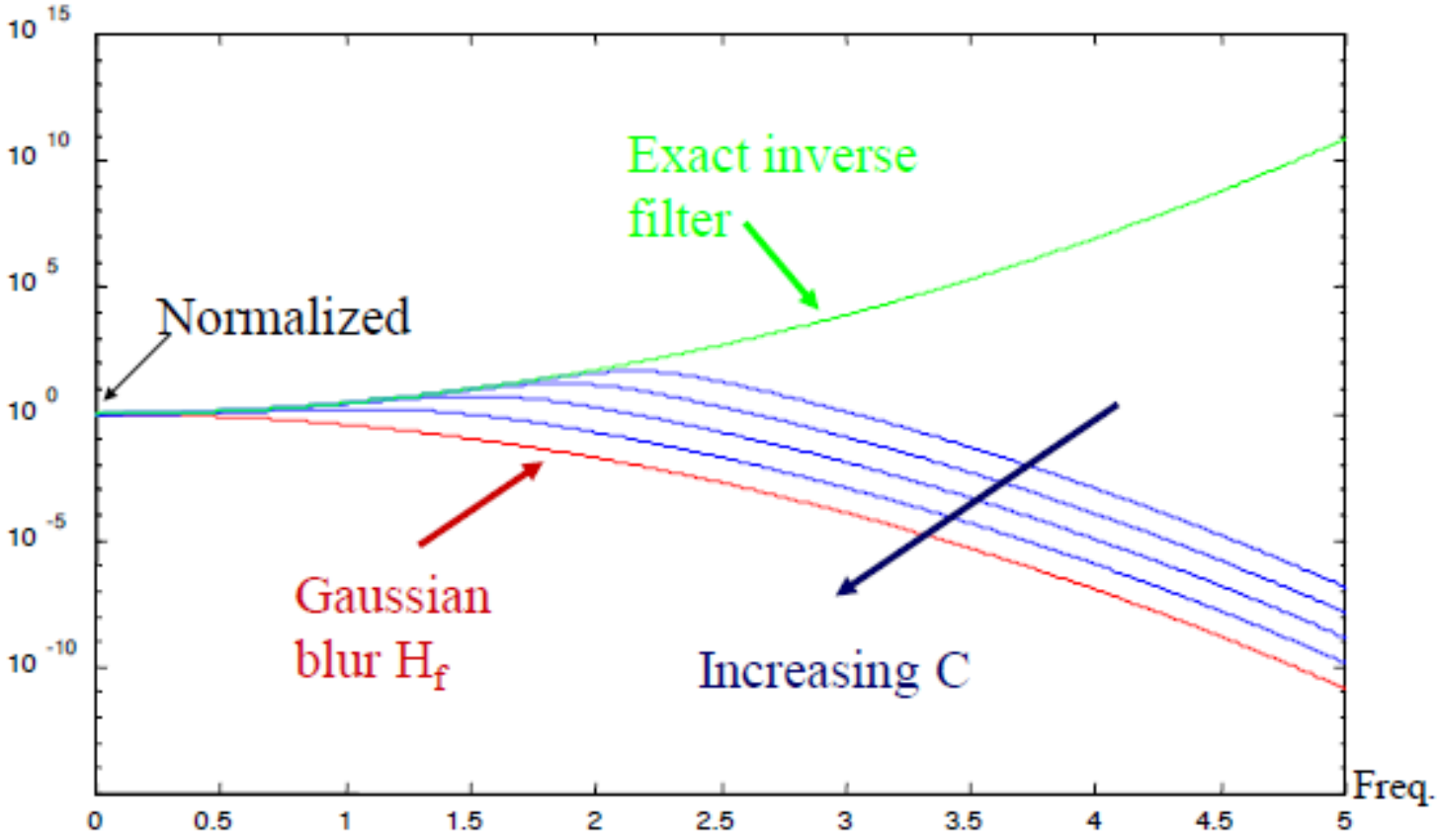
Avoiding Singularities

Note: If H has zero elements, then at those frequencies, the inverse filter does not exist. Thus use the following inverting equation instead:

$$\frac{H_f^*}{H_f^* H_f + C} \approx \begin{cases} H_f^{-1} & \text{for } |H_f|^2 >> C \\ \frac{1}{C} H_f^* & \text{for } |H_f|^2 << C \end{cases}$$

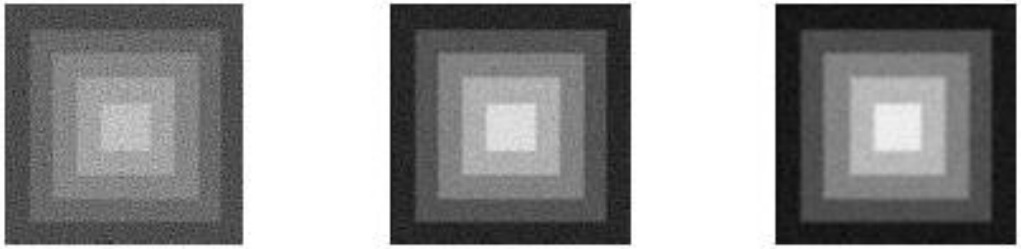
Scales image by 1/C at high frequencies

A Simple Example

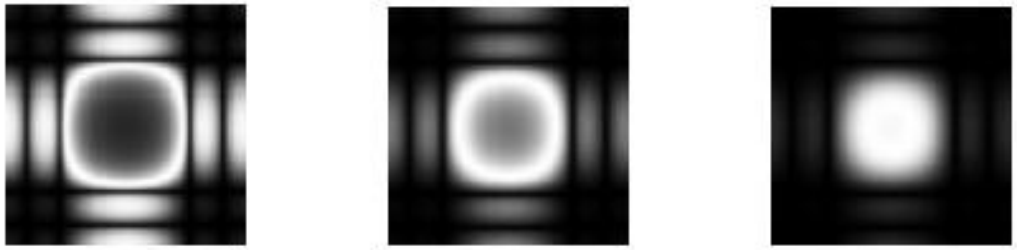


Restoration Example

Restored f :



Inverse Filter Used:



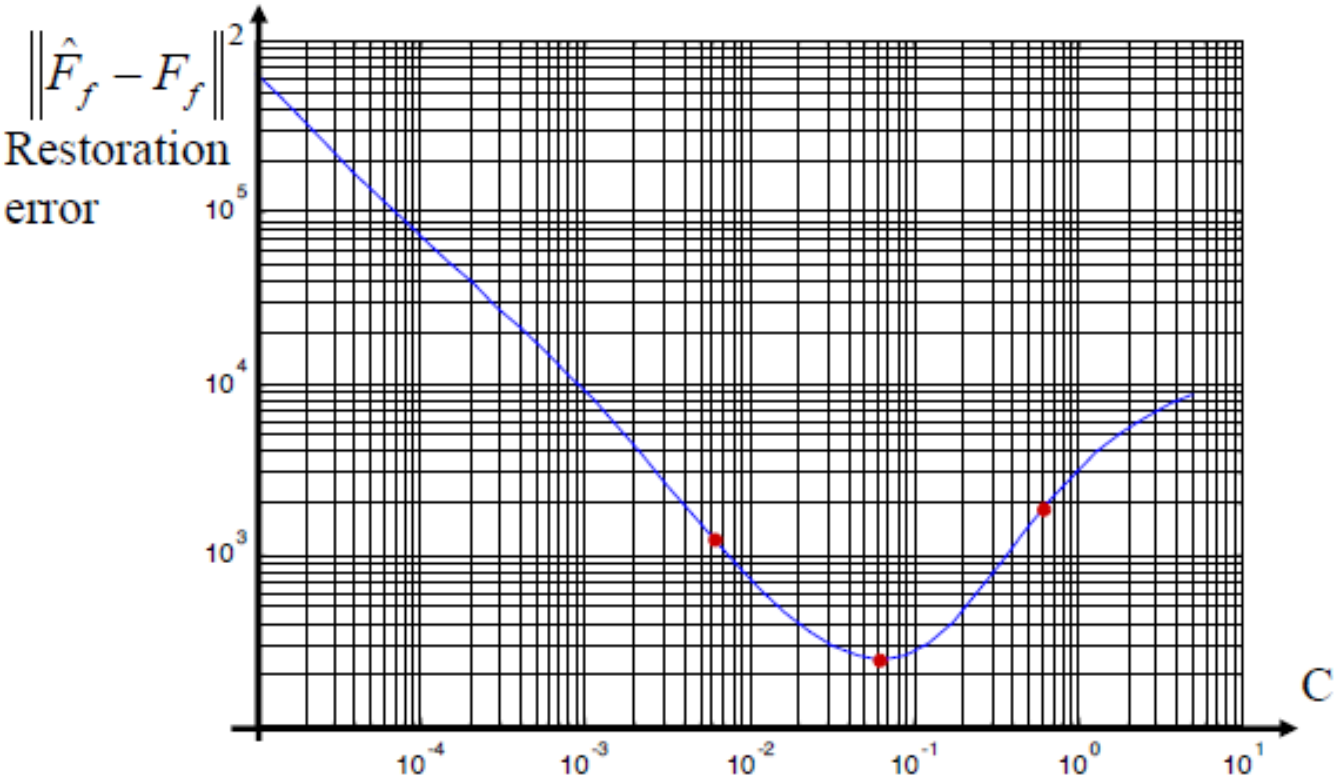
C=0.0063 C=0.063 C=0.63

$\left\|\hat{F}_f - F_f\right\|^2 \longrightarrow$ Error=1264 Error=247 Error=2046

Larger C suppresses high frequency response in the inverse filter

WHAT IS THE BEST CHOICE FOR C?

Choice of the Parameter C



This is not a practical way of finding the best C !!

Drawbacks of this approach

- Finding the best C is not simple.
- In the proposed strategy, C is constant for all frequencies!! Maybe we can gain something by using a frequency dependent value.
- For high frequencies where the noise is dominant (over the signal), the image is amplified by 1/C>>1. This is silly, since we amplify mostly noise!

A Different Strategy

Previous Strategy - Based on H_f:
Where H_f is high, apply exact inverse, and where low, apply H_f/C

New Startegy - Based on [Signal/Noise]*H_f :
$|H_f|$SNR>>1 - inverse
$|H_f|$SNR<<1 - 0

$$\text{Signal/Noise} = \text{SNR}^2 = \frac{\left|F(\omega_x, \omega_y)\right|^2}{\sigma^2}$$

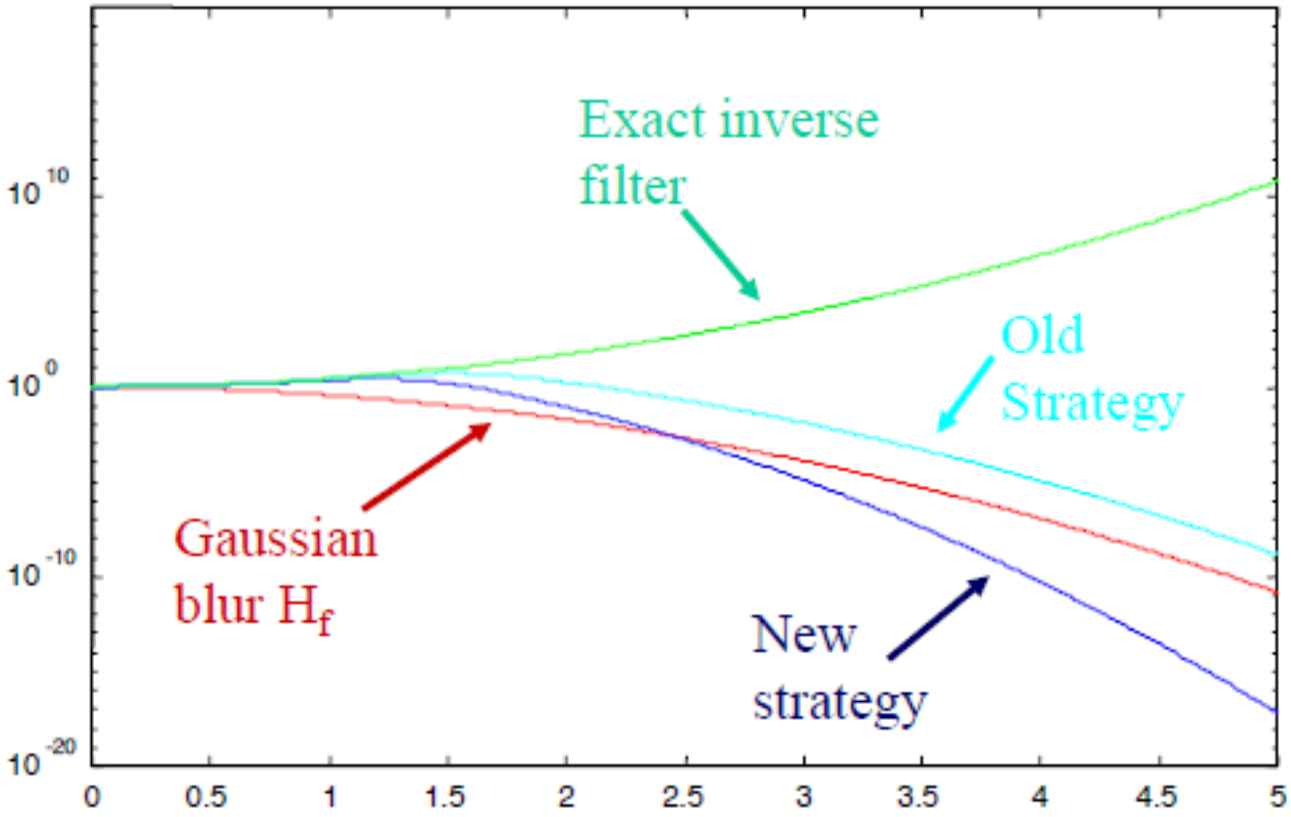
The Wiener Filter

SNR is used in the following manner:

$$\frac{H_f^*}{H_f^* H_f + C \cdot SNR^{-2}} =$$

$$= \begin{cases} H_f^{-1}, & \text{when} \quad |H_f|^2 \cdot SNR^2 >> C \\ \frac{SNR^2}{C} H_f^*, & \text{when} \quad |H_f|^2 \cdot SNR^2 << C \end{cases}$$

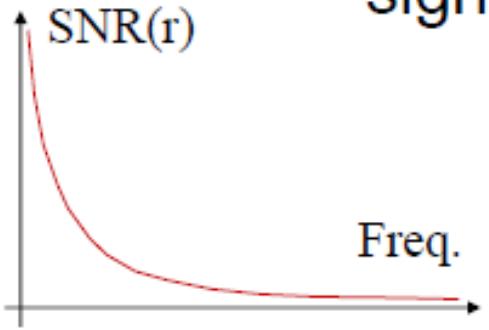
Simple Example



Choice of SNR

SNR is not known!

Assume white noise with unit variance & radialy decaying signal:

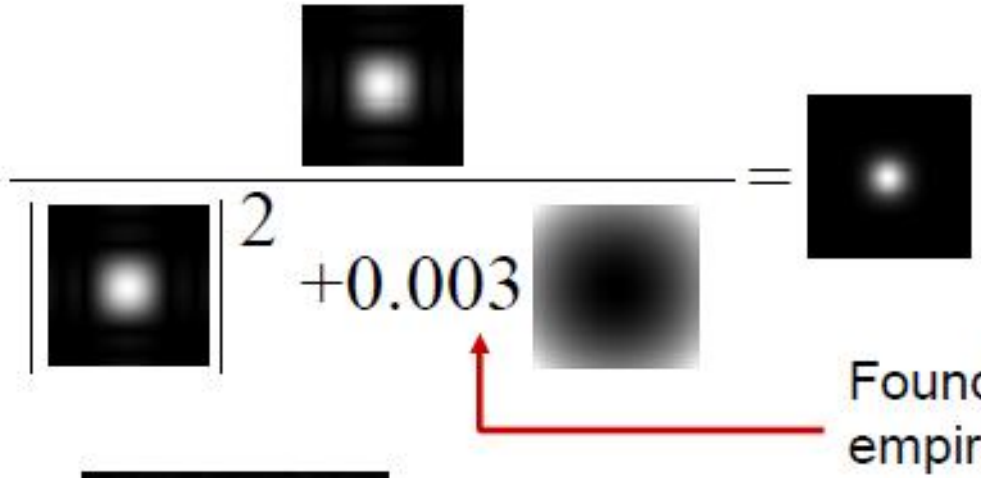


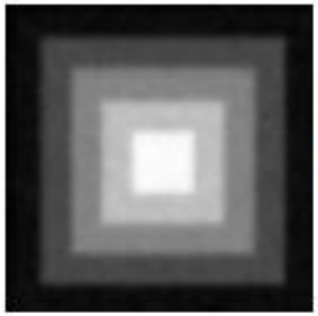
$$N^2(r,\theta)=1$$

$$F^2(r,\theta)=r^{-\rho} \quad \rho \approx 2$$

Polar Coordinates in Freq. Domain

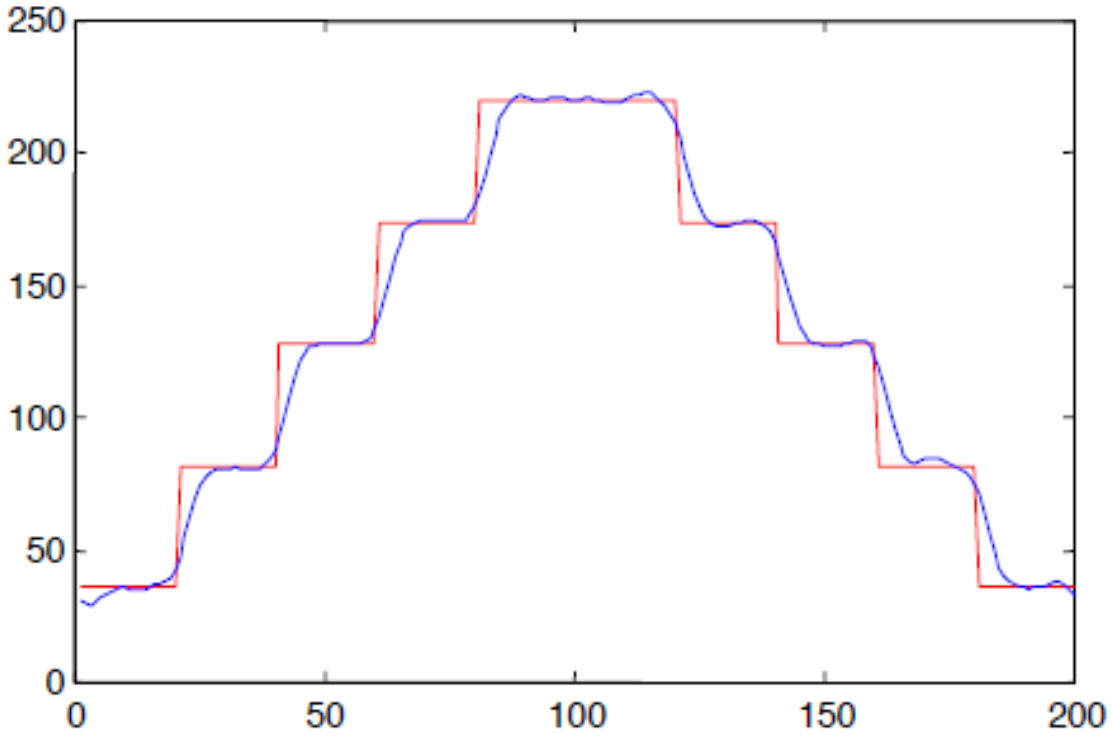
Restoration Example





Error=106 (compared to the 247 in the previous approach)

Is this the best one can do?



NO !

Cross-Correlation and Cross Spectrum

- Cross-correlation function for discrete space (stationary) random signals.

$$C_{fg}(l,m) = \mathrm{E}\left[f(x+l, y+m)g^{*}(x,y)\right]$$

- Cross-correlation function for discrete space deterministic signals.

$$C_{fg}(l,m) = \frac{1}{LM}\sum_{x,y} f(x+l, y+m)g^{*}(x,y)$$

: xcorr2.m

- Cross power spectrum

Fourier Transform

$$C_{fg}(l,m) \Longleftrightarrow C_{fg}(\omega_x, \omega_y) = F(\omega_x, \omega_y)G^{*}(\omega_x, \omega_y)$$

Auto-Correlation and Power Spectral Density (PSD)

- Auto-correlation function for discrete space (stationary) random signals.

$$C_{ff}(l,m) = \mathrm{E}\left[f(x+l, y+m) f^{*}(x,y)\right]$$

- Auto-correlation function for discrete space deterministic signals.

$$C_{ff}(l,m) = \frac{1}{LM} \sum_{x,y} f(x+l, y+m) f^{*}(x,y)$$

- (Auto) power spectrum

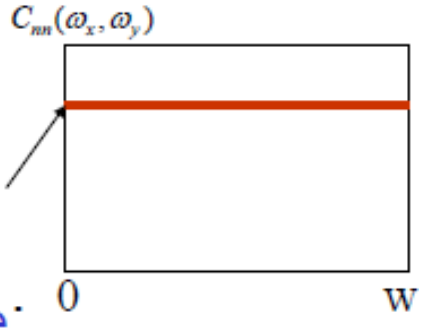
Fourier Transform

$$C_{ff}(l,m) \Longleftrightarrow C_{ff}(\omega_x, \omega_y) = F^{*}(\omega_x, \omega_y) F(\omega_x, \omega_y) = |F|^2$$

Power Spectrum of White and Colored Noise

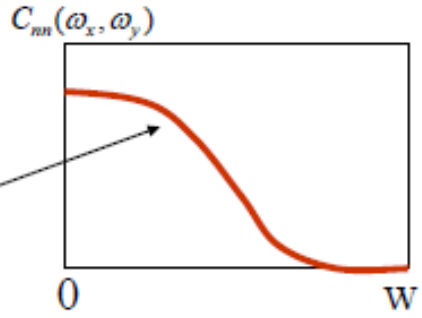
- PSD for White (uncorrelated) Noise:

$$C_{nn}(\omega_x,\omega_y) = FT\left[\mathrm{E}\left(n(x+l,y+m)n(x,y)\right)\right]$$
$$= FT\left[v^2\delta(l,m)\right] = v^2 FT\left[\delta(l,m)\right] = v^2$$

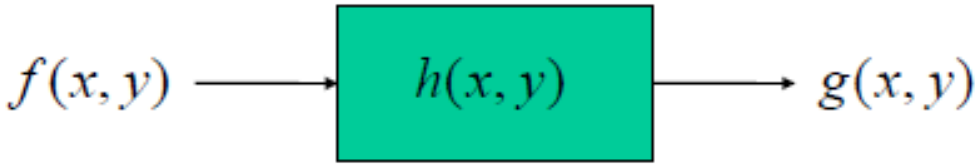


- PSD for Colored (correlated) Noise:

$$C_{nn}(\omega_x,\omega_y) = FT\left[\mathrm{E}\left(n(x+l,y+m)n(x,y)\right)\right]$$
$$= FT\left[v^2\rho(l,m)\right] = v^2 FT\left[\rho(l,m)\right]$$
$$= v^2 R(\omega_x,\omega_y)$$



Power Spectra and Linear Systems: Basic Rules



$$C_{gg}(\omega_x,\omega_y) = \left|H(\omega_x,\omega_y)\right|^2 C_{ff}(\omega_x,\omega_y)$$

PSD of Output

PSD of Input

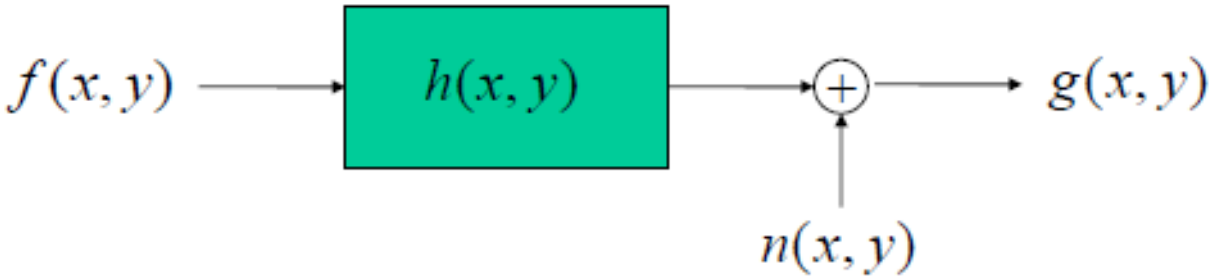
$$C_{gf}(\omega_x,\omega_y) = H(\omega_x,\omega_y)C_{ff}(\omega_x,\omega_y)$$

$$C_{fg}(\omega_x,\omega_y) = H^*(\omega_x,\omega_y)C_{ff}(\omega_x,\omega_y)$$

Cross PSD of Output with Input

Cross PSD of Input with Output

Power Spectra and Linear Systems



$$C_{gg}(\omega_x,\omega_y) = \left|H(\omega_x,\omega_y)\right|^2 C_{ff}(\omega_x,\omega_y) + C_{nn}(\omega_x,\omega_y)$$

PSD of Output

PSD of Input

PSD of noise

Power Spectra and Linear Systems: Example:

$$C_{gg}(\omega_x,\omega_y) = \left|H(\omega_x,\omega_y)\right|^2 C_{ff}(\omega_x,\omega_y) + C_{nn}(\omega_x,\omega_y)$$

$$f(x,y) = \sqrt{2\pi}e^{-2\pi^2(x^2+y^2)} \Rightarrow C_{ff}(\omega_x,\omega_y) = \left|F(\omega_x,\omega_y)\right|^2 = e^{-(\omega_x{}^2+\omega_y{}^2)}$$

$$h(x,y) = \sqrt{2\pi}\sigma e^{-2\pi^2\sigma^2(x^2+y^2)} \Rightarrow \left|H(\omega_x,\omega_y)\right|^2 = e^{-\frac{\omega_x{}^2+\omega_y{}^2}{\sigma^2}}$$

$$n(x,y) \sim \mathrm{N}(0,\nu^2) \Rightarrow$$

$$C_{nn}(\omega_x,\omega_y) = FT\left[\mathrm{E}\left(n(x+m,y+n)n(x,y)\right)\right]$$

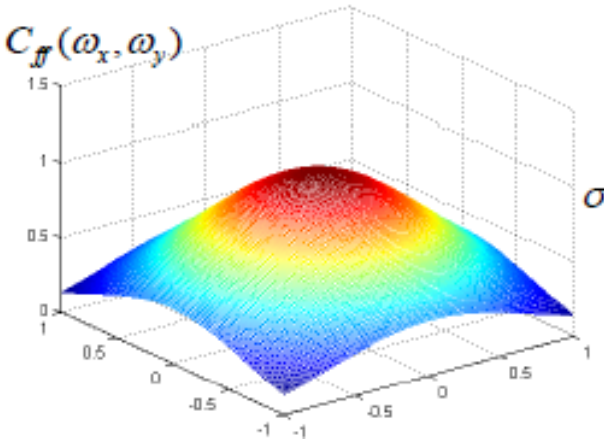
$$= FT\left[\nu^2\delta(m,n)\right] = \nu^2 FT\left[\delta(m,n)\right] = \nu^2$$

Power Spectra and Linear Systems: Example:

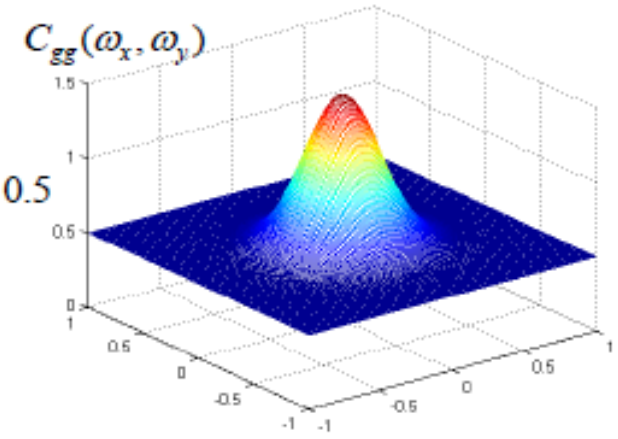
$$C_{gg}(\omega_x,\omega_y) = \left|H(\omega_x,\omega_y)\right|^2 C_{ff}(\omega_x,\omega_y) + C_{nn}(\omega_x,\omega_y)$$

$$C_{gg}(\omega_x,\omega_y) = e^{-(\omega_x^2+\omega_y^2)} e^{-\frac{\omega_x^2+\omega_y^2}{\sigma^2}} + \nu^2$$

$$= e^{-\left(\frac{1+\sigma^2}{\sigma^2}\right)(\omega_x^2+\omega_y^2)} + \nu^2$$



$\sigma^2 = 0.1, \quad \nu^2 = 0.5$



Statistical Properties of the Stabilized Inverse Filter

$$H_r(\omega_x,\omega_y)=\frac{H^*}{\left|H(\omega_x,\omega_y)\right|^2+c}$$

$$\hat{F}(\omega_x,\omega_y)=\frac{H^*G}{|H|^2+c}=\frac{H^*(HF+N)}{|H|^2+c}=\frac{|H|^2F+H^*N}{|H|^2+c}=\frac{|H|^2F}{|H|^2+c}+\frac{H^*N}{|H|^2+c}$$

$$\mathrm{E}\left[\hat{F}\right]=\frac{|H|^2F}{|H|^2+c} \quad \Rightarrow \quad Bias=F-\mathrm{E}\left[\hat{F}\right]=F-\frac{|H|^2F}{|H|^2+c}=\left(\frac{c}{|H|^2+c}\right)F$$

Make c small

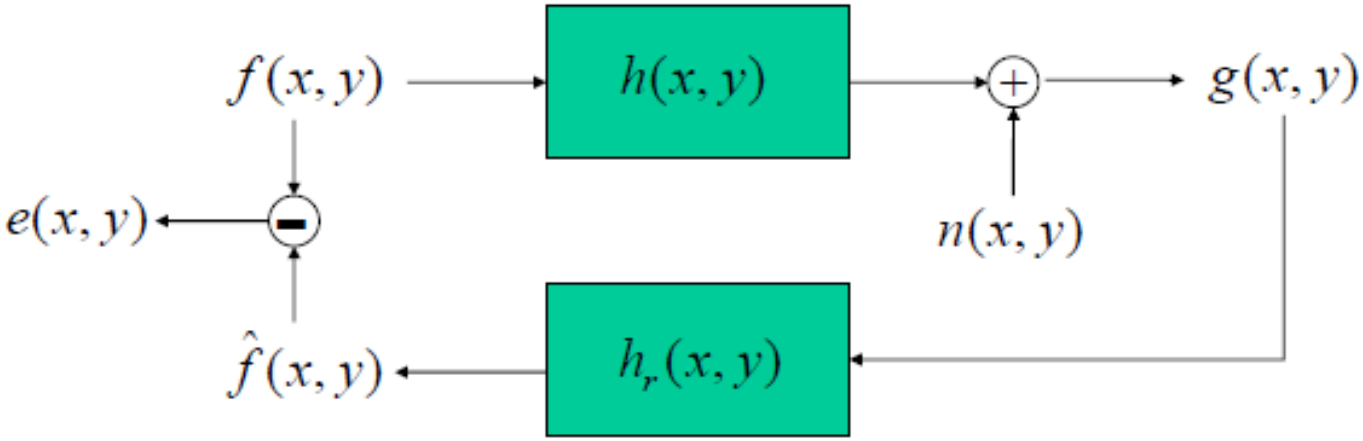
Want an estimator of the image f which has small bias and small variance!

$$C_{\hat{f}\hat{f}}=\frac{|H|^2\nu^2}{\left(|H|^2+c\right)^2}$$

Make c large

Conflict

Wiener Filter: Minimum Mean Squared Error



Design a restoration filter such that the reconstruction error is as small as possible

$$\min_{h_r(x,y)} \mathrm{E}\left\| \underbrace{f(x,y)-\hat{f}(x,y)}_{e(x,y)} \right\|^2 \equiv \min_{h_r(x,y)} C_{ee}(\omega_x,\omega_y)$$

Parseval's Relation

Wiener Filter: Minimum Mean Squared Error

$$C_{ee}(\omega_x,\omega_y) = C_{\tilde{f}\tilde{f}}(\omega_x,\omega_y) + C_{ff}(\omega_x,\omega_y) - C_{f\hat{f}}(\omega_x,\omega_y) - C_{\hat{f}f}(\omega_x,\omega_y)$$

$$\begin{aligned} C_{\tilde{f}\tilde{f}}(\omega_x,\omega_y) &= \left|H_r(\omega_x,\omega_y)\right|^2 C_{gg}(\omega_x,\omega_y) \\ &= \left|H_r(\omega_x,\omega_y)\right|^2 \left(\left|H(\omega_x,\omega_y)\right|^2 C_{ff}(\omega_x,\omega_y) + C_{nn}(\omega_x,\omega_y)\right) \end{aligned}$$

$$C_{f\hat{f}}(\omega_x,\omega_y) = C_{gf}(\omega_x,\omega_y) H_r^{*}(\omega_x,\omega_y)$$

$$C_{\hat{f}f}(\omega_x,\omega_y) = C_{fg}(\omega_x,\omega_y) H_r(\omega_x,\omega_y)$$

Wiener Filter: Minimum Mean Squared Error

For each frequency (ω_x, ω_y) can minimize the PSD $C_{ee}(\omega_x, \omega_y)$

$$\frac{\partial C_{ee}}{\partial H_r} = 0 \quad \Rightarrow \quad H_r(\omega_x, \omega_y) = \frac{C_{fg}(\omega_x, \omega_y)}{\left|H(\omega_x, \omega_y)\right|^2 C_{ff}(\omega_x, \omega_y) + C_{nn}(\omega_x, \omega_y)}$$

$$= \frac{C_{ff}(\omega_x, \omega_y) H^*(\omega_x, \omega_y)}{\left|H(\omega_x, \omega_y)\right|^2 C_{ff}(\omega_x, \omega_y) + C_{nn}(\omega_x, \omega_y)}$$

$$= \frac{H^*(\omega_x, \omega_y)}{\left|H(\omega_x, \omega_y)\right|^2 + \dfrac{C_{nn}(\omega_x, \omega_y)}{C_{ff}(\omega_x, \omega_y)}}$$

SNR^{-2}

Wiener Filter Example:

$$f(x,y) = \sqrt{2\pi}e^{-2\pi^2(x^2+y^2)}$$
$$h(x,y) = \sqrt{2\pi}\sigma e^{-2\pi^2\sigma^2(x^2+y^2)}$$
$$n(x,y) \sim \mathrm{N}(0,\nu^2)$$

$$H_r(\omega_x,\omega_y) = \frac{H^*(\omega_x,\omega_y)}{\left|H(\omega_x,\omega_y)\right|^2 + \frac{C_{nn}(\omega_x,\omega_y)}{C_{ff}(\omega_x,\omega_y)}}$$

$$= \frac{e^{-\frac{\omega_x^2+\omega_y^2}{2\sigma^2}}}{e^{-\frac{\omega_x^2+\omega_y^2}{\sigma^2}} + \nu^2 e^{(\omega_x^2+\omega_y^2)}}$$

Minimum Mean Square Error Filtering (Wiener Filtering)

- Assumption:
  - Noise and image are uncorrelated.
  - One or the other has zero mean.
  - The gray levels in the estimate are a linear function of the levels in the degradation image.

Wiener Filtering

a b c

FIGURE 5.28 Comparison of inverse- and Wiener filtering. (a) Result of full inverse filtering of Fig. 5.25(b). (b) Radially limited inverse filter result. (c) Wiener filter result.

Full inverse filtering

Radially limited inverse filtering

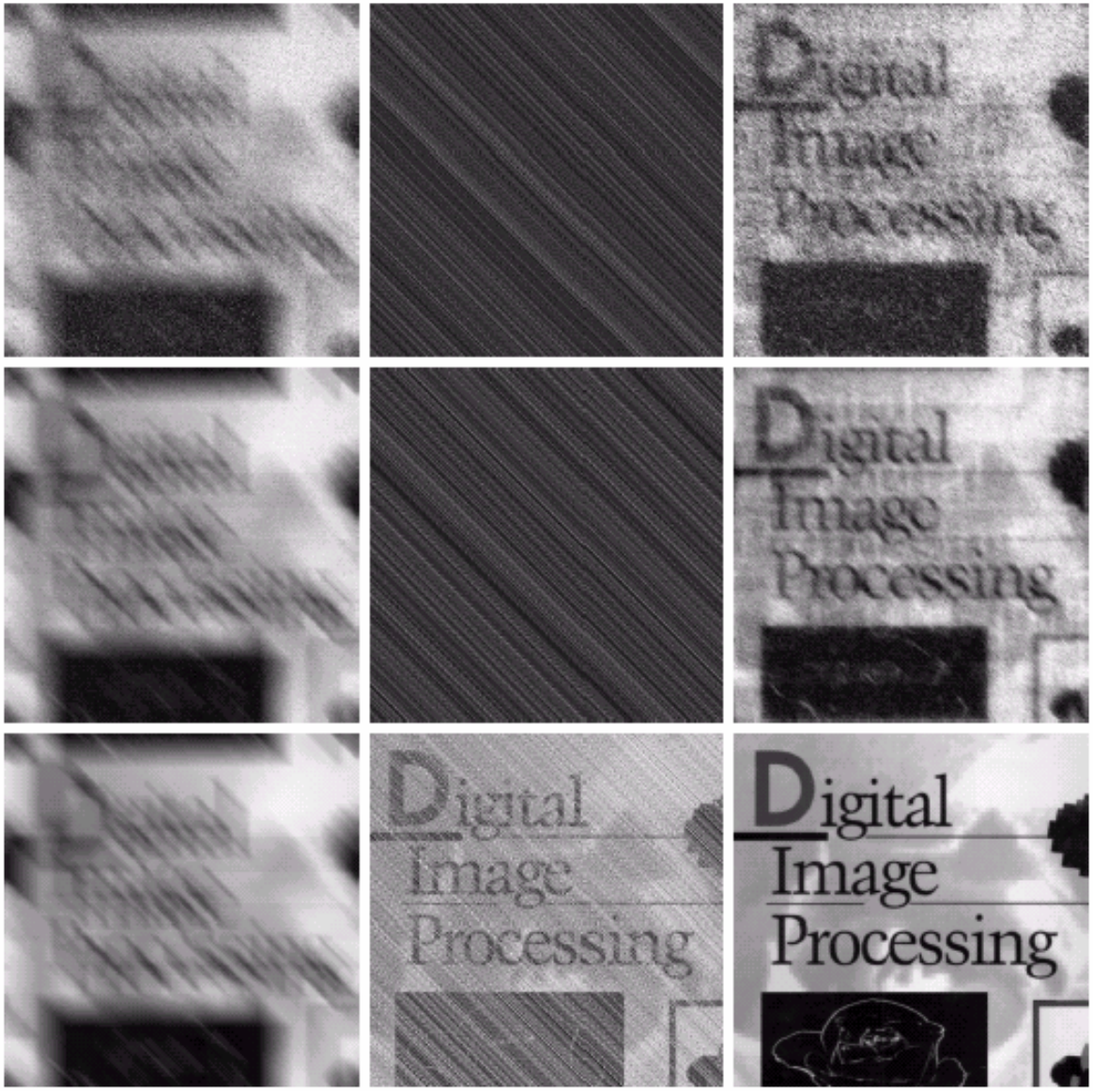
Wiener filtering

Wiener Filtering

Inverse filtering

Wiener filtering

Reduced noise variance



Wiener Filter Special Cases:

$$H_W(\omega_x,\omega_y)=\frac{H^*(\omega_x,\omega_y)}{\left|H(\omega_x,\omega_y)\right|^2+\frac{C_{nn}(\omega_x,\omega_y)}{C_{ff}(\omega_x,\omega_y)}}$$

Denoising: No Blur (H=1)

$$H_r(\omega_x,\omega_y)=\frac{1}{1+\frac{C_{nn}(\omega_x,\omega_y)}{C_{ff}(\omega_x,\omega_y)}}=\frac{C_{ff}(\omega_x,\omega_y)}{C_{ff}(\omega_x,\omega_y)+C_{nn}(\omega_x,\omega_y)}$$

Deblurring: No Noise (C_{nn}=0)

$$H_r(\omega_x,\omega_y)=\frac{H^*(\omega_x,\omega_y)}{\left|H(\omega_x,\omega_y)\right|^2}=\frac{1}{H(\omega_x,\omega_y)}$$ → Inverse Filter!

Wiener Filter Useful Facts:

$$H_W(\omega_x,\omega_y)=\frac{H^*(\omega_x,\omega_y)}{\left|H(\omega_x,\omega_y)\right|^2+\dfrac{C_{nn}(\omega_x,\omega_y)}{C_{ff}(\omega_x,\omega_y)}}$$

Phase: Has same phase response as Inverse Filter

$$\text{Phase}[H_W(\omega_x,\omega_y)]=\text{Phase}[H^*(\omega_x,\omega_y)]$$

$$=-\text{Phase}[H(\omega_x,\omega_y)]=\text{Phase}[\frac{1}{H(\omega_x,\omega_y)}]$$

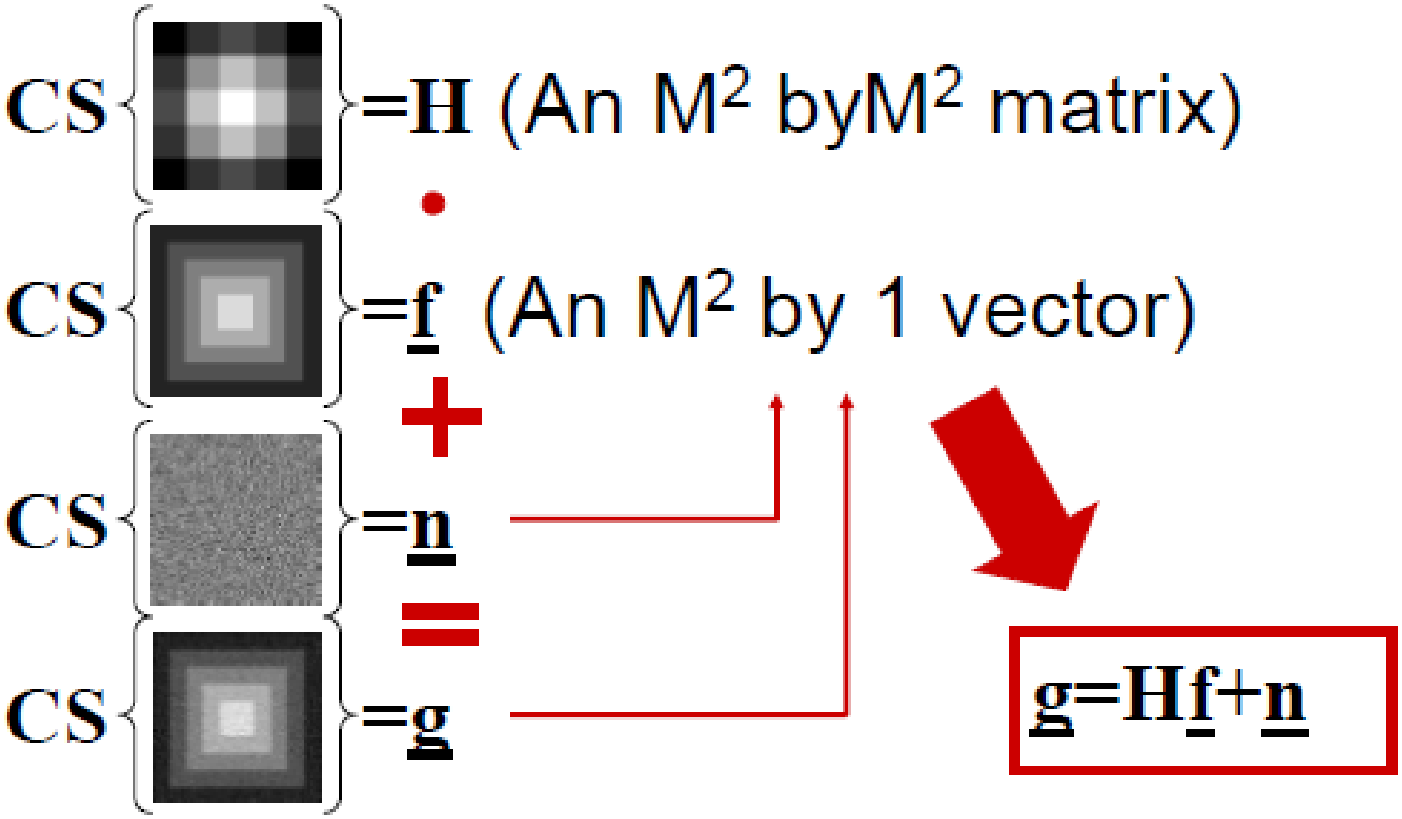
Practicality: Need to specify C_{nn} and C_{ff}

For example: $C_{nn}(\omega_x,\omega_y)=\nu^2$ $C_{ff}(\omega_x,\omega_y)=\left(\omega_x^2+\omega_y^2\right)^{-1}$ "Natural" images

Guess, estimate, from sensor

Not the same as knowing f !

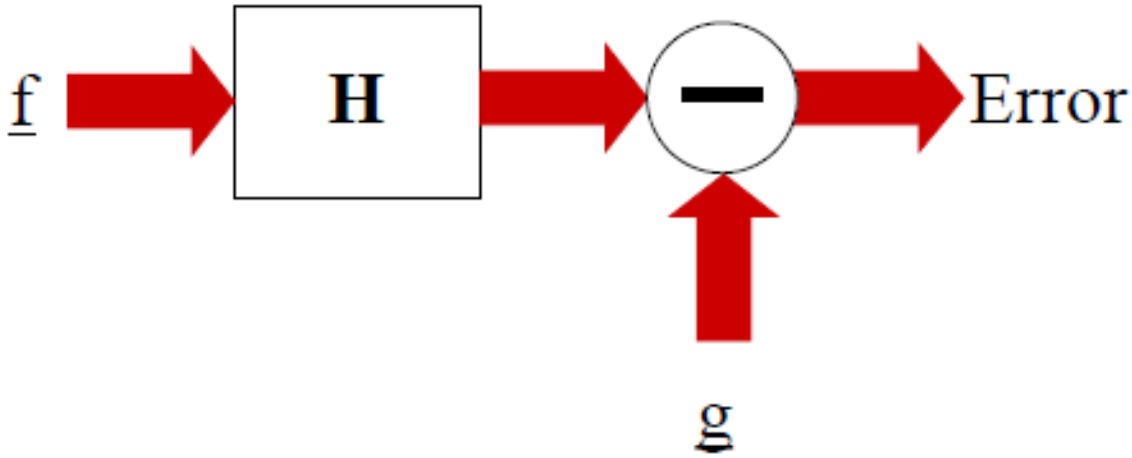
The Forward Model: Matrix Formulation



Maximum-Likelihood/Least Squares

$$\underline{\mathbf{g}} = \mathbf{H}\underline{\mathbf{f}} + \underline{\mathbf{n}} \quad \Rightarrow \quad \text{Assume Gaussian noise:}$$

$$\hat{\underline{\mathbf{f}}}_{ML} = \underset{\underline{\mathbf{f}}}{ArgMin} \ \varepsilon^2_{ML}(\underline{\mathbf{f}}) = \underset{\underline{\mathbf{f}}}{ArgMin} \left\| \underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}} \right\|^2$$



Solving the ML

$$\underline{\mathbf{g}} = \mathbf{H}\underline{\mathbf{f}} + \underline{\mathbf{n}} \quad \Rightarrow$$

$$\hat{\underline{\mathbf{f}}}_{ML} = \underset{\underline{\mathbf{f}}}{ArgMin} \ \varepsilon^2_{ML}(\underline{\mathbf{f}}) = \underset{\underline{\mathbf{f}}}{ArgMin} \left\| \underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}} \right\|^2$$

$$= \underset{\underline{\mathbf{f}}}{ArgMin} \left(\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}}\right)^{\mathbf{T}} \left(\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}}\right)$$

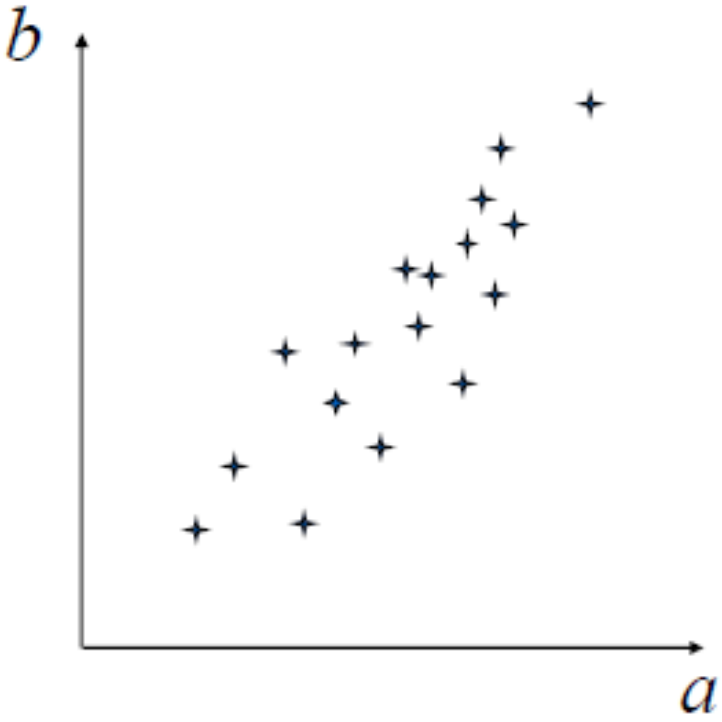
$$\Rightarrow \quad \frac{\partial \boldsymbol{\varepsilon}^{\mathbf{2}}(\underline{\mathbf{f}})}{\partial \underline{\mathbf{f}}} = -2\mathbf{H}^{\mathbf{T}}\left[\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}}\right] \Rightarrow \mathbf{H}^{\mathbf{T}}\underline{\mathbf{g}} = \mathbf{H}^{\mathbf{T}}\mathbf{H}\underline{\mathbf{f}}$$

$$\Rightarrow \quad \hat{\underline{\mathbf{f}}}_{\mathbf{ML}} = \left[\mathbf{H}^{\mathbf{T}}\mathbf{H}\right]^{-1}\mathbf{H}^{\mathbf{T}}\underline{\mathbf{g}} \quad \text{(LeastSquares)}$$

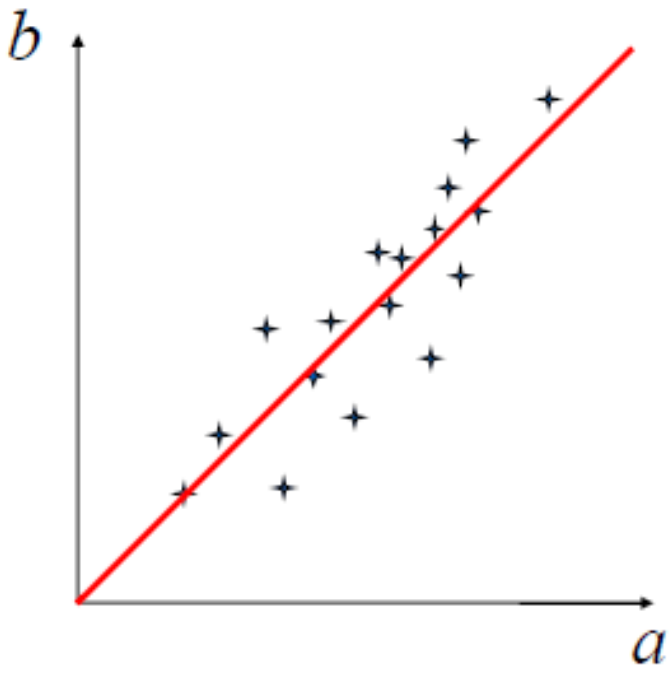
$\mathbf{H}$ square and non-singular: $\hat{\underline{\mathbf{f}}}_{\mathbf{ML}} = \mathbf{H}^{-1}\underline{\mathbf{g}}$

Ordinary Least-Squares

One-dimensional regression



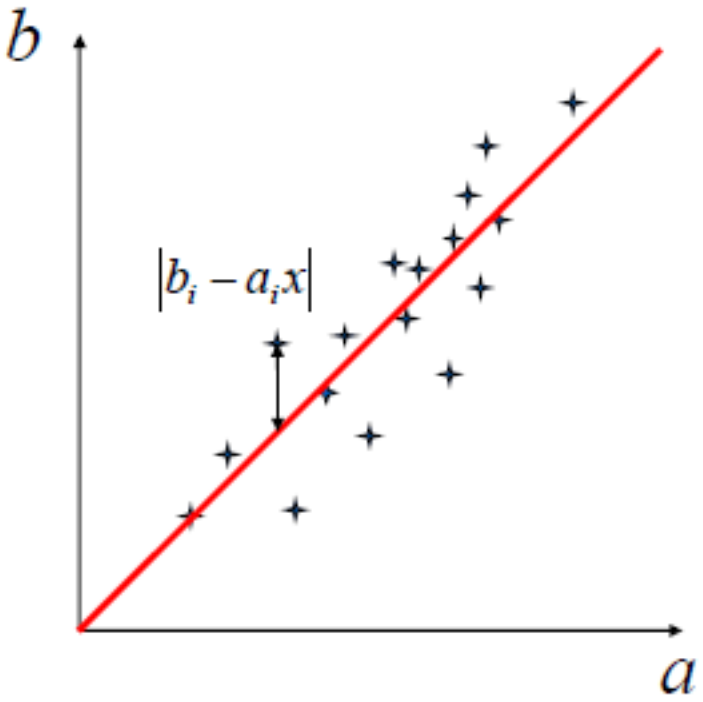
One-dimensional regression



Find a line that represent the "best" linear relationship:

$$b = ax$$

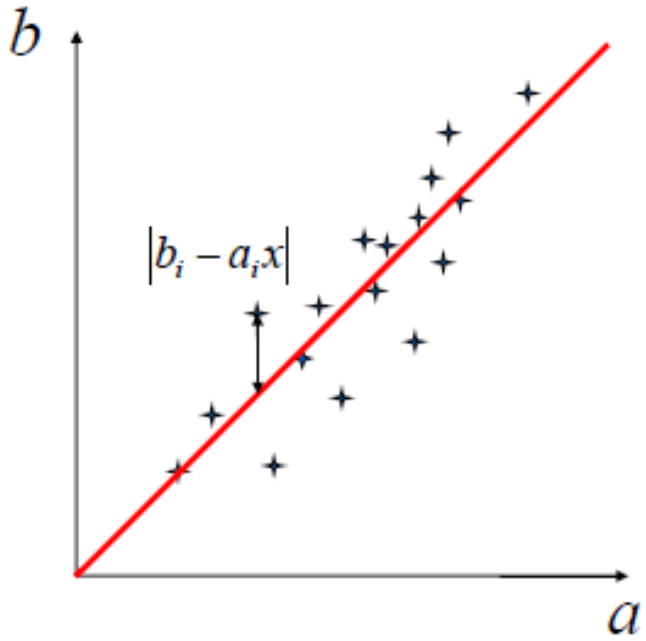
One-dimensional regression



- Problem: the data does not go through a line

$$e_i = b_i - a_i x$$

One-dimensional regression



- Problem: the data does not go through a line

$$e_i = b_i - a_i x$$

- Find the line that minimizes the sum:

$$\sum_i (b_i - a_i x)^2$$

- We are looking for $\hat{x}$ that minimizes

$$e(x) = \sum_i (b_i - a_i x)^2$$

Matrix notation

Using the following notations

$$\boldsymbol{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \quad \text{and} \quad \boldsymbol{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$

we can rewrite the error function using linear algebra as:

$$e(x) = \sum_i (b_i - a_i x)^2$$

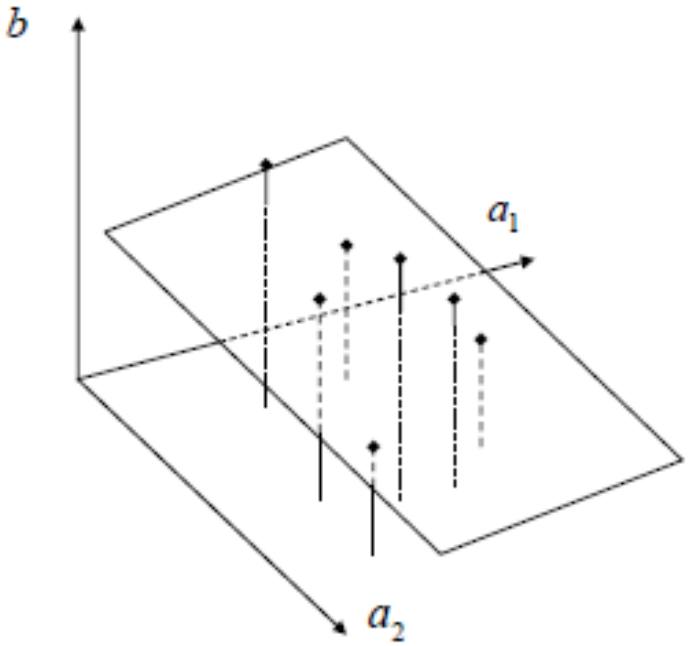
$$= (\boldsymbol{b} - x\boldsymbol{a})^T (\boldsymbol{b} - x\boldsymbol{a})$$

$$e(x) = \left\| \boldsymbol{b} - x\boldsymbol{a} \right\|^2$$

Multidimentional linear regression

Using a model with *m* parameters

$$b = a_1 x_1 + \ldots + a_m x_m = \sum_j a_j x_j$$



Multidimentional linear regression

Using a model with *m* parameters

$$b = a_1 x_1 + \ldots + a_m x_m = \sum_j a_j x_j$$

and *n* measurements

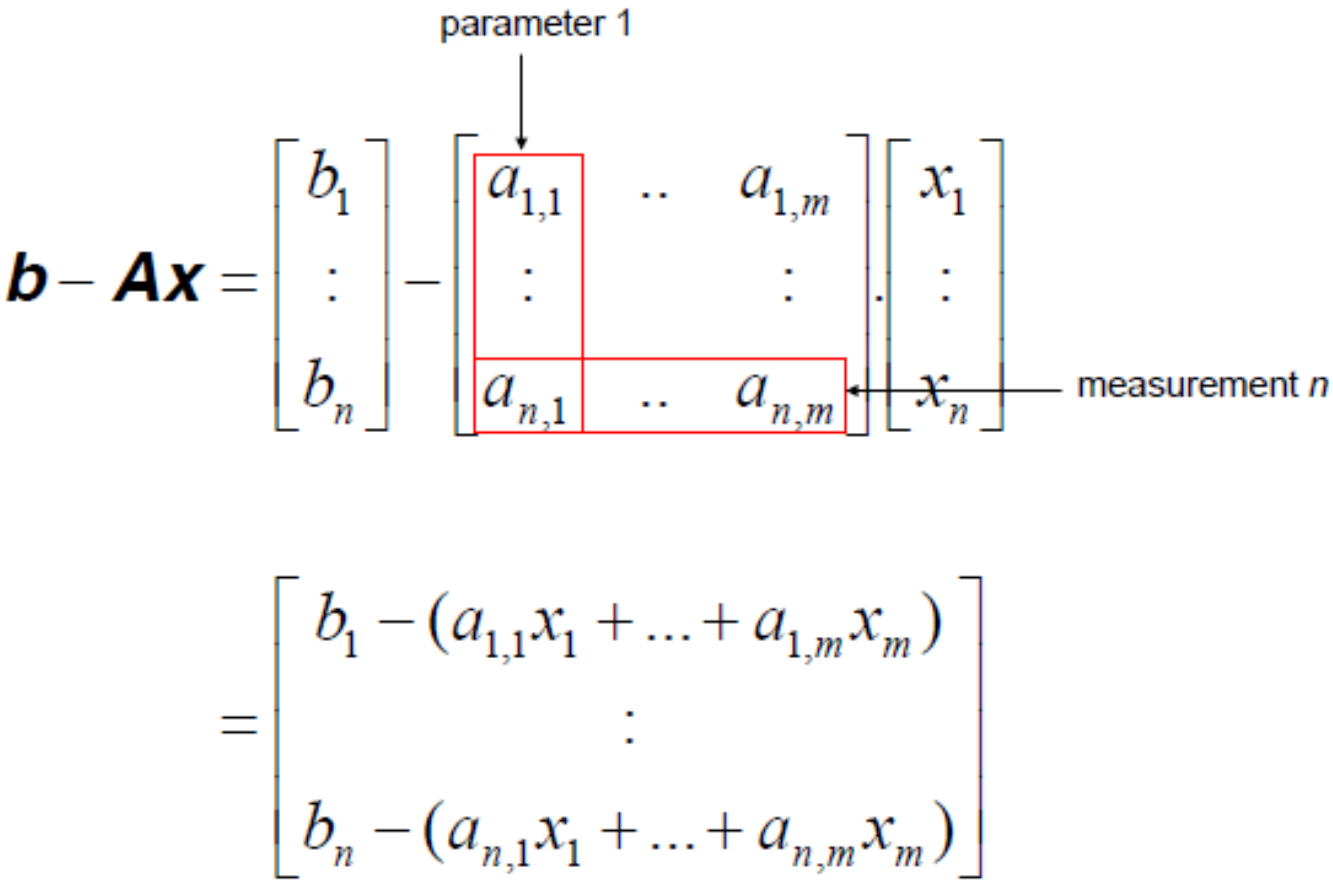
$$e(\boldsymbol{x}) = \sum_{i=1}^{n} (b_i - \sum_{j=1}^{m} a_{i,j} x_j)^2$$

$$= \left\| \boldsymbol{b} - \left[\sum_{j=1}^{m} a_{i,j} x_j \right] \right\|^2$$

$$e(\boldsymbol{x}) = \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|^2$$

$$\boldsymbol{b} - \boldsymbol{Ax} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} - \begin{bmatrix} a_{1,1} & .. & a_{1,m} \\ \vdots & & \vdots \\ a_{n,1} & .. & a_{n,m} \end{bmatrix} . \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$$

$$= \begin{bmatrix} b_1 - (a_{1,1}x_1 + \ldots + a_{1,m}x_m) \\ \vdots \\ b_n - (a_{n,1}x_1 + \ldots + a_{n,m}x_m) \end{bmatrix}$$


parameter 1
$\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} - \begin{bmatrix} a_{1,1} & .. & a_{1,m} \\ \vdots & & \vdots \\ a_{n,1} & .. & a_{n,m} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$
measurement n
$= \begin{bmatrix} b_1 - (a_{1,1}x_1 + \ldots + a_{1,m}x_m) \\ \vdots \\ b_n - (a_{n,1}x_1 + \ldots + a_{n,m}x_m) \end{bmatrix}$

Minimizing

$$e(\boldsymbol{x}) = \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|^2$$

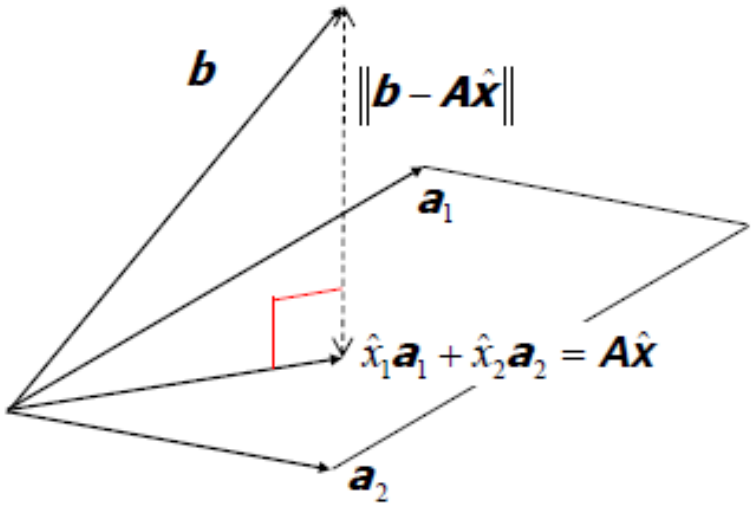
$\hat{\boldsymbol{x}}$ minimizes $e(\boldsymbol{x})$ if

- $\boldsymbol{A}^T\boldsymbol{A}\hat{\boldsymbol{x}} = \boldsymbol{A}^T\boldsymbol{b}$
  The ***normal equation***
- $\boldsymbol{A}^T\boldsymbol{A}$ is positive semi-definite
  Always true

Geometric interpretation

- $A\hat{x}$ is the orthogonal projection of b onto *range*(A)

$$\Leftrightarrow A^T(b - A\hat{x}) = 0 \Leftrightarrow A^T A\hat{x} = A^T b$$



Recap:

$$\underline{\mathbf{g}} = \mathbf{H}\underline{\mathbf{f}} + \underline{\mathbf{n}} \quad \Rightarrow \quad \underline{\hat{\mathbf{f}}}_{ML} = \underset{\underline{\mathbf{f}}}{ArgMin} \left\| \underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}} \right\|^2$$

$$= \underset{\underline{\mathbf{f}}}{ArgMin} \left(\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}} \right)^{\mathbf{T}} \left(\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}} \right)$$

$$\Rightarrow \quad \frac{\partial \boldsymbol{\varepsilon}^2(\underline{\mathbf{f}})}{\partial \underline{\mathbf{f}}} = -2\mathbf{H}^{\mathbf{T}}\left[\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}} \right] \Rightarrow \mathbf{H}^{\mathbf{T}}\underline{\mathbf{g}} = \mathbf{H}^{\mathbf{T}}\mathbf{H}\underline{\mathbf{f}}$$

$$\Rightarrow \quad \underline{\hat{\mathbf{f}}}_{\mathbf{ML}} = \left[\mathbf{H}^{\mathbf{T}}\mathbf{H} \right]^{-1} \mathbf{H}^{\mathbf{T}}\underline{\mathbf{g}} \quad \text{(LeastSquares)}$$

Regularization

If **H** is (close to) singular, we can apply the following "stabilized" equation instead:

$$\hat{\underline{f}}_{MAP} = \left[\mathbf{H}^T\mathbf{H} + \lambda \cdot \mathbf{I}\right]^{-1}\mathbf{H}^T\underline{\mathbf{g}}$$

Small constant — Identity matrix

Equivalently.....

$$\varepsilon^2_{MAP}(\underline{\mathbf{f}}) = \left\|\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}}\right\|^2 + \lambda\left\|\underline{\mathbf{f}}\right\|^2$$

i.e., prefer solutions with smaller norm

Regularization

$$\varepsilon^2_{MAP}(\underline{\mathbf{f}}) = \left\|\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}}\right\|^2 + \lambda\left\|\underline{\mathbf{f}}\right\|^2 \quad \text{differentiate.....}$$

$$\frac{\partial \boldsymbol{\varepsilon}^2(\underline{\mathbf{f}})}{\partial \underline{\mathbf{f}}} = -2\mathbf{H}^{\mathbf{T}}\left[\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}}\right] + 2\lambda\underline{\mathbf{f}} = 0$$

$$\Rightarrow \left[\mathbf{H}^{\mathbf{T}}\mathbf{H} + \lambda \cdot \mathbf{I}\right]\underline{\mathbf{f}} = \mathbf{H}^{\mathbf{T}}\underline{\mathbf{g}}$$

$$\underline{\hat{\mathbf{f}}}_{\mathbf{MAP}} = \left[\mathbf{H}^{\mathbf{T}}\mathbf{H} + \lambda \cdot \mathbf{I}\right]^{-1}\mathbf{H}^{\mathbf{T}}\underline{\mathbf{g}}$$

Why regularization?

- We have seen that

$$\min_x \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|^2 \text{ has a unique solution}$$

$$\Leftrightarrow$$

the columns of $\boldsymbol{A}$ form an independert system

- But what happens if the system is almost singular?
  - The solution becomes very sensitive to the data
  - Poor conditioning

The 1-dimensional case

- The 1-dimensional normal equation

$$\boldsymbol{a}^T \boldsymbol{a}\hat{x} = \boldsymbol{a}^T \boldsymbol{b}$$

$$\text{if } \|\boldsymbol{a}\| \neq \boldsymbol{0} \Rightarrow \hat{x} = \frac{\boldsymbol{a}^T \boldsymbol{b}}{\boldsymbol{a}^T \boldsymbol{a}}$$

$$\text{if } \|\boldsymbol{a}\| \approx \boldsymbol{0} \Rightarrow \text{trouble}$$

The Singular Value Decomposition

- Every matrix $\boldsymbol{A}$ (nxm) can be decomposed into:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^{\mathrm{T}}$$

  - where
    - $\boldsymbol{U}$ is an nxn orthogonal matrix
    - $\boldsymbol{V}$ is an mxm orthogonal matrix
    - $\boldsymbol{S}$ is an nxm diagonal matrix

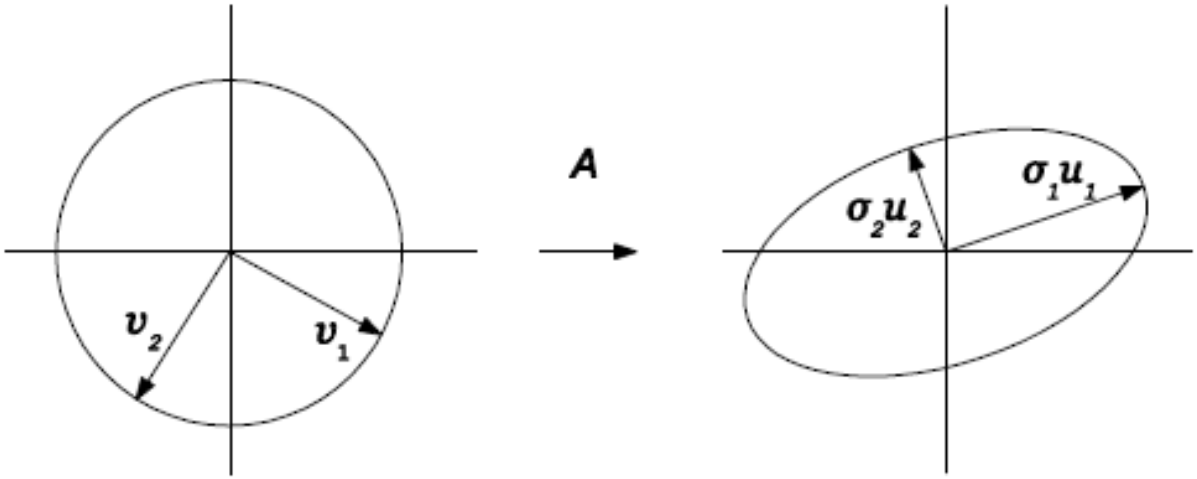
The Singular Value Decomposition

- Every matrix ***A*** (*nxm*) can be decomposed into:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^{\mathrm{T}}$$

$$\boldsymbol{A} = \begin{bmatrix} u_{1,1} & .. & u_{1,n} \\ : & & : \\ u_{n,1} & .. & u_{n,n} \end{bmatrix} \cdot \begin{bmatrix} \sigma_1 & .. & 0 \\ : & & : \\ 0 & .. & \sigma_m \\ : & & : \\ 0 & .. & 0 \end{bmatrix} \cdot \begin{bmatrix} v_{1,1} & .. & v_{1,m} \\ : & & : \\ v_{m,1} & .. & v_{m,m} \end{bmatrix}^{T}$$

Geometric interpretation



$$A = \begin{bmatrix} u_{1,1} & .. & u_{1,n} \\ : & & : \\ u_{n,1} & .. & u_{n,n} \end{bmatrix} \begin{bmatrix} \sigma_1 & .. & 0 \\ : & & : \\ 0 & .. & \sigma_m \\ : & & : \\ 0 & .. & 0 \end{bmatrix} \begin{bmatrix} v_{1,1} & .. & v_{1,m} \\ : & & : \\ v_{m,1} & .. & v_{m,m} \end{bmatrix}^T$$

Solving with the SVD

$$A^T A\hat{x} = A^T b \quad \text{and} \quad A = USV^T$$

$$\Rightarrow \left(VS^T U^T USV^T\right)\hat{x} = VS^T U^T b$$

$$\Rightarrow \left(S^T SV^T\right)\hat{x} = S^T U^T b$$

$$\Rightarrow \hat{x} = \left(S^T SV^T\right)^{-1} S^T U^T b = V\left(S^T S\right)^{-1} S^T U^T b$$

$$\hat{x} = VS^{+}U^T b = A^{+}b = \left(A^T A\right)^{-1} A^T b$$

Pseudo-inverse

Solving with the SVD

$$A^T A\hat{x} = A^T b \quad \text{and} \quad A = USV^T$$

$$\hat{x} = \left(S^T S\right)^{-1} S^T U^T b = VS^{+}U^T b$$

$$\hat{x} = \begin{bmatrix} v_{1,1} & .. & v_{1,m} \\ : & & : \\ v_{m,1} & .. & v_{m,m} \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{\sigma_1} & .. & 0 \\ : & & : \\ 0 & .. & \frac{1}{\sigma_m} \\ : & & : \\ 0 & .. & 0 \end{bmatrix} \cdot \begin{bmatrix} u_{1,1} & .. & u_{1,n} \\ : & & : \\ u_{n,1} & .. & u_{n,n} \end{bmatrix}^T b$$

Solving with the SVD

$$A^T A\hat{x} = A^T b \quad \text{and} \quad A = USV^T$$

$$\hat{x} = \left(S^T S\right)^{-1} S^T U^T b = VS^{+}U^T b$$

$$\hat{x} = \begin{bmatrix} v_1 & .. & v_m \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{\sigma_1} & .. & 0 \\ : & & : \\ 0 & .. & \frac{1}{\sigma_m} \\ : & & : \\ 0 & .. & 0 \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ : \\ u_n \end{bmatrix} b = \sum_{i=1}^{m} \frac{1}{\sigma_i} v_i u_i^T b$$

A is nearly rank defficient

$$\boldsymbol{A} = \begin{bmatrix} u_{1,1} & .. & u_{1,n} \\ : & & : \\ u_{n,1} & .. & u_{n,n} \end{bmatrix} \begin{bmatrix} \sigma_1 & .. & 0 \\ : & & : \\ 0 & .. & \approx 0 \\ : & & : \\ 0 & .. & 0 \end{bmatrix} \begin{bmatrix} v_{1,1} & .. & v_{1,m} \\ : & & : \\ v_{m,1} & .. & v_{m,m} \end{bmatrix}^T$$

Condition number measures closeness to singularity:

$$Cond(\mathrm{A}) = \frac{\sigma_{\max}}{\sigma_{\min}}$$

$$(\text{good}) \leftarrow \quad 1 \leq Cond(\mathrm{A}) \leq \infty \quad \rightarrow (\text{bad})$$

A is nearly rank defficient

- A is nearly rank defficient
  - some of the σ_i are close to 0
  - some of the $\frac{1}{\sigma_i}$ are close to ∞

Problem with:

$$\hat{\boldsymbol{x}} = \sum_{i=1}^{m} \frac{1}{\sigma_i} \boldsymbol{v}_i \boldsymbol{u}_i^T \boldsymbol{b}$$

Possible Solution: Truncated SVD – little control over smoothness

with $\sigma_1 \geq \cdots \geq \sigma_m$ and $\hat{\boldsymbol{x}} = \sum_{i=1}^{k<m} \frac{1}{\sigma_i} \boldsymbol{v}_i \boldsymbol{u}_i^T \boldsymbol{b}$

Damped least-squares

- Replace

$$\min_x \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|^2$$

by

$$\min_x \left(\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|^2 + \lambda \|\boldsymbol{D}\boldsymbol{x}\|^2 \right)$$

where λ is a scalar and $\boldsymbol{D}$ is a matrix

The solution $\hat{\boldsymbol{x}}$ verifies

$$\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{D}^T\boldsymbol{D}\right)\hat{\boldsymbol{x}} = \boldsymbol{A}^T\boldsymbol{b}$$

Examples of $\boldsymbol{D}$

Diagonal

$$\boldsymbol{D} = \begin{bmatrix} d_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & d_n \end{bmatrix}$$

Differential

$$\boldsymbol{D} = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}$$

Limit scale

$$\hat{\boldsymbol{x}} = \sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + l_i^2} \boldsymbol{v}_i \boldsymbol{u}_i^T \boldsymbol{b}$$

Enforce smoothness

$$\hat{\boldsymbol{x}} = \sum_{i=1}^{n} f(\sigma_i)\ \boldsymbol{v}_i \boldsymbol{u}_i^T \boldsymbol{b}$$

where $f(\sigma_i) \rightarrow 0$

Better MAP Prior

$$\varepsilon^2_{MAP}(\underline{\mathbf{f}}) = \left\|\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}}\right\|^2 + \lambda\left\|\mathbf{D}\underline{\mathbf{f}}\right\|^2$$

Among the solutions, find those that are relatively smooth

(**D** - Laplacian derivative $\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$)

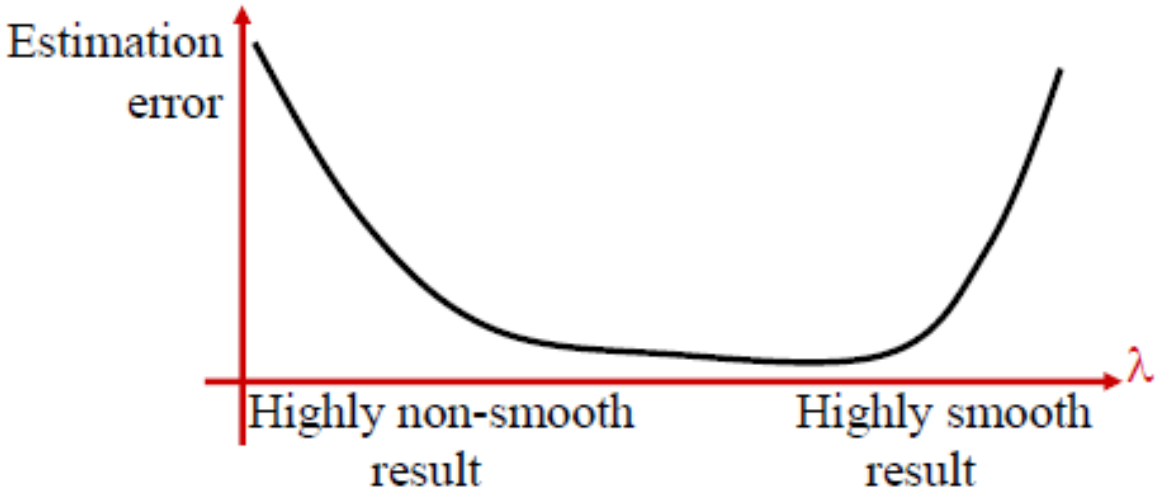
$$\hat{\underline{\mathbf{f}}}_{MAP} = \left[\mathbf{H}^T\mathbf{H} + \lambda \cdot \mathbf{D}^T\mathbf{D}\right]^{-1}\mathbf{H}^T\underline{\mathbf{g}}$$

Choice of Lambda

$$\varepsilon^2_{MAP}(\underline{\mathbf{f}}) = \left\|\underline{\mathbf{g}} - \mathbf{H}\underline{\mathbf{f}}\right\|^2 + \lambda\left\|\mathbf{D}\underline{\mathbf{f}}\right\|^2$$

Measurement Error

Smoothness Error



Interpretations?

$$\hat{\underline{f}}_{ML} = \mathbf{H}^{-1}\underline{g}$$

→ Inverse filter

$$\hat{\underline{f}}_{MAP} = \left[\mathbf{H}^T\mathbf{H} + \lambda \cdot \mathbf{I}\right]^{-1}\mathbf{H}^T\underline{g}$$

→ Heuristic (Stabilized Inverse) approach

$$\hat{\underline{f}}_{MAP} = \left[\mathbf{H}^T\mathbf{H} + \lambda \cdot \mathbf{D}^T\mathbf{D}\right]^{-1}\mathbf{H}^T\underline{g}$$

→ ≈Wiener filter