Image Restoration & Reconstruction

Instructor: Yiming Xiao

Email: yiming.xiao@concordia.ca

Department of Computer Science
and Software Engineering
Concordia University

Materials adapted from Dr. T. D. Bui

Image Restoration

Image Enhancement : make image look better

Image Restoration : image has been degraded by
- noise
- blur
- motion

$f(x,y)$ → *Degraded* → $\oplus$ → $g(x,y)$

$h(x,y)$ *noise*

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

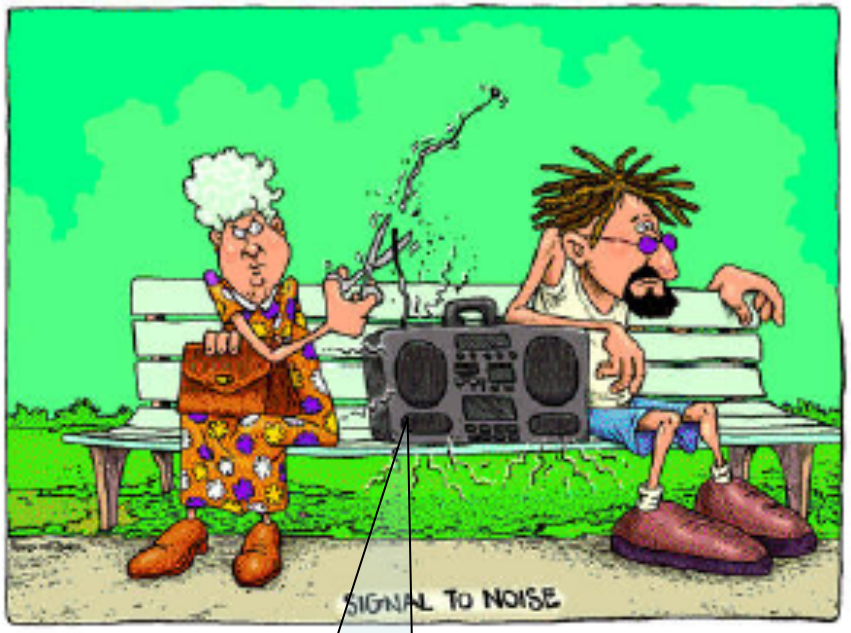
Restoration : $g(x,y)$ → Restoration filter → $\hat{f}(x,y)$

Criteria : Minimize $E[(f(x,y) - \hat{f}(x,y))^2]$

First we treat $h(x, y) = 1$, *i.e.* only corruption is noise :

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

Assumption : Noise is uncorrelated w.r.t. image itself.



Noise or signal?
may depend on
whom you ask

Image Denoising

Left: Noisy image (PSNR=18.64 dB); Right: denoised image (PSNR=30.71dB).

What is a Decibel?

Suppose we have two loudspeakers, the first playing a sound with ***power P_1****, and another playing a louder version of the same sound with* ***power P_2****, but everything else kept the same.*

The difference in decibels between the two is defined to be :

$10 \log_{10} (P_2/P_1)$ *dB*

$P_1 = P_2 \Rightarrow 0\,\mathrm{dB}$

If P_2 is twice P_1, the difference in *dB* is

$10 \log_{10} (P_2/P_1) = 10 \log_{10} 2 =$ **3 *dB***.

If the second has 10 times the power of the first, the difference in *dB* would be:
$10 \log_{10} (P_2/P_1) = 10 \log_{10} 10 =$ 10 *dB*.

If the second has a million times the power of the first, the difference in *dB* would be: $10 \log_{10} (P_2/P_1) = 10 \log_{10} 1000000 =$ 60 dB.

What is Noise?

The two main limitations in image accuracy are **blur** and **noise**.

Blur: intrinsic to image acquisition systems, as digital images have a finite number of samples.

Noise: each pixel value $u(i)$ is the result of a light intensity measurement, usually made by the element of a sensor. When the light source is constant, the number of photons/signals received by each pixel fluctuates around its average.

In a first rough approximation one can write $\boldsymbol{v(i) = u(i) + n(i);}$

$\boldsymbol{v(i)}$ the observed value,
$\boldsymbol{u(i)}$ the true value at pixel i,
$\boldsymbol{n(i)}$ the noise.

- The amount of noise is signal-dependent, that is $n(i)$ is larger when $u(i)$ is larger.
- In noise models, the normalized values of $n(i)$ and $n(j)$ at different pixels are assumed to be independent random variables (white noise).

Signal-to-Noise Ratio

A good quality photograph (for visual inspection) has about 256 gray level values, where 0 represents black and 255 represents white. Measuring the amount of noise by its standard deviation, $\sigma(n)$ one can define the signal to noise ratio (SNR) as:

$$SNR = 10 \log_{10} \frac{\sigma^2(u)}{\sigma^2(n)} \quad \text{dB}$$

where $\sigma^2(u)$ denotes the empirical variance of *u*(*i*),

$$\sigma^2(u) = \frac{1}{|I|} \sum_i (u(i) - \bar{u})^2$$

and $\bar{u} = \frac{1}{|I|} \sum_{i \in I} u(i)$ is the average gray level value.

The variance of the noise can also be obtained as an empirical measurement, or formally computed when the noise model and parameters are known.

A good quality image has a standard deviation of about 60 (or $\sigma^2 = 3600$).

One way to test the effect of noise on a standard digital image is to add a Gaussian white noise, in which case n(i) are i.i.d. Gaussian real variables.

Signal-to-Noise Ratio

➡ When $\sigma(n) = 3$, no visible noise effect is observed..

Thus, for a $SNR = 10\log_{10}\frac{\sigma^2(u)}{\sigma^2(n)} = 10\log_{10}\frac{3600}{9} = 26$ the noise is nearly invisible.

Surprisingly, one can add white noise up to 2/1 ratio and still *see* everything in a picture. It justifies the many attempts to find convincing denoising algorithms.

➡ Denoising algorithms see no difference between small details and noise, and therefore remove them. In many cases, they create new distortions and researchers refer to them as artifacts: ringing, blur, staircase effect….

➡ In experimental settings, the noise model is perfectly precise. So the weak point of the algorithms is the inadequacy of the image model. All of the methods assume that the noise is oscillatory, and that the image is smooth, or piecewise smooth.

- *Noise is not necessarily only in high frequency*
- *Structural details can be in high spatial frequency*
- *The smoothness assumption for an image doesn't work all the time*

Signal-to-Noise Ratio

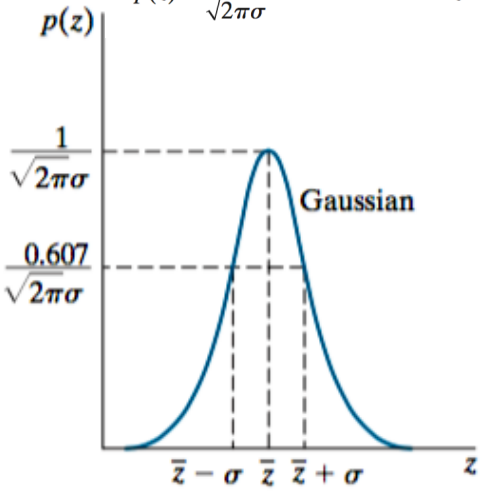
(a) a digital image with standard deviation $\sigma(u) = 55$, (b) the same image with noise added (standard deviation $\sigma(n) = 3$), the signal-to-noise ratio being therefore equal to 25.26 dB, (c) the same image with signal-to-noise ratio slightly larger than 6 dB.

Courtesy of Buades, Coll, and Morel

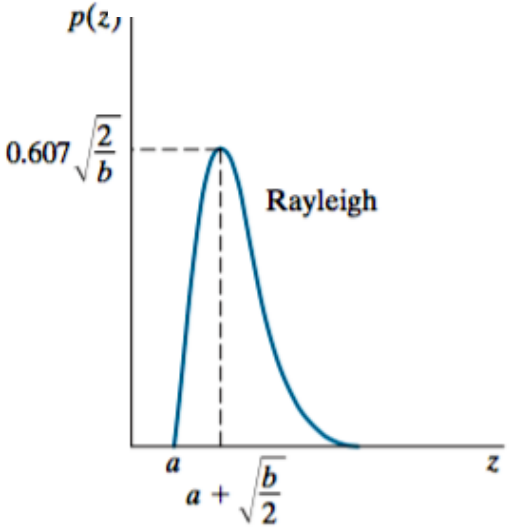
$$PSNR = 20\log_{10}\left(\frac{255}{\sigma(n)}\right)$$

Noise models

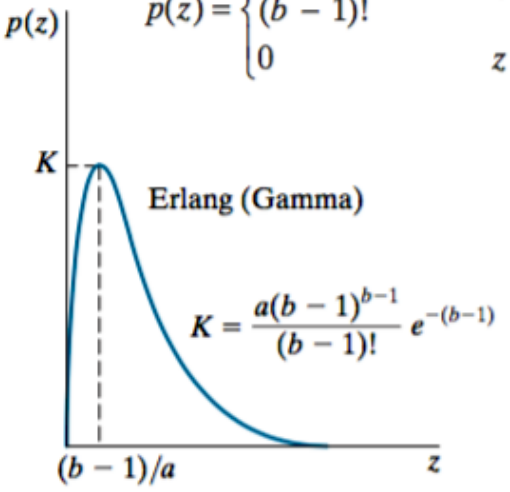
$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(z-\bar{z})^2}{2\sigma^2}} \qquad -\infty < z < \infty$$

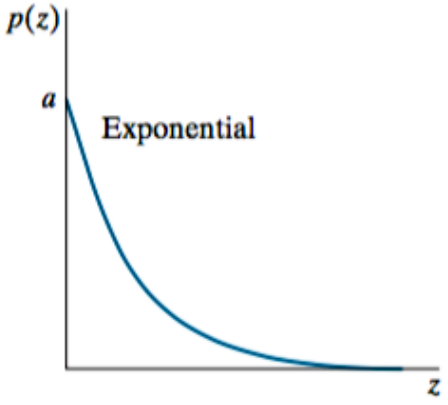


$$p(z) = \begin{cases} \frac{2}{b}(z-a)e^{-(z-a)^2/b} & z \geq a \\ 0 & z < a \end{cases}$$

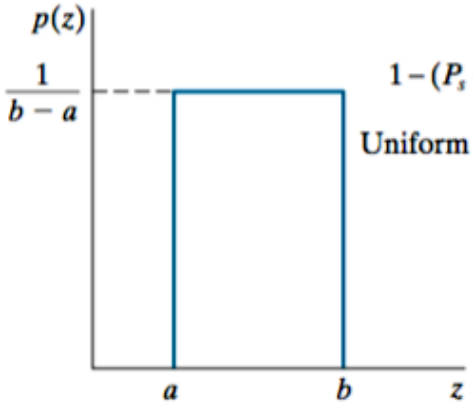


$$p(z) = \begin{cases} \frac{a^b z^{b-1}}{(b-1)!} e^{-az} & z \geq 0 \\ 0 & z < 0 \end{cases}$$

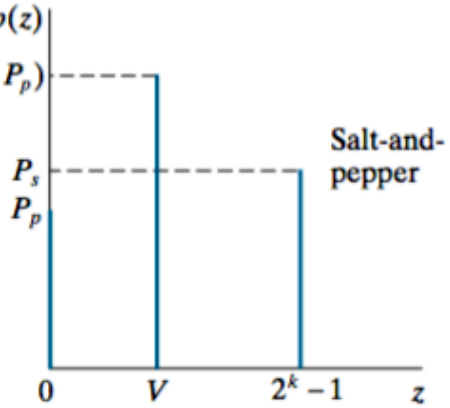




$$p(z) = \begin{cases} ae^{-az} & z \geq 0 \\ 0 & z < 0 \end{cases}$$



$$p(z) = \begin{cases} \frac{1}{b-a} & a \leq z \leq b \\ 0 & \text{otherwise} \end{cases}$$



$$p(z) = \begin{cases} P_s & \text{for } z = 2^k - 1 \\ P_p & \text{for } z = 0 \\ 1-(P_s+P_p) & \text{for } z = V \end{cases}$$

Noise models

Gaussian noise:

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(z-\bar{z})^2}{2\sigma^2}} \qquad -\infty < z < \infty$$

Convenient modeling in many cases

Rayleigh noise:

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

Great for estimating histograms with skewness; background noise in MRI

Salt and pepper noise:

$$p(z) = \begin{cases} P_s & \text{for } z = 2^k - 1 \\ P_p & \text{for } z = 0 \\ 1-(P_s+P_p) & \text{for } z = V \end{cases}$$

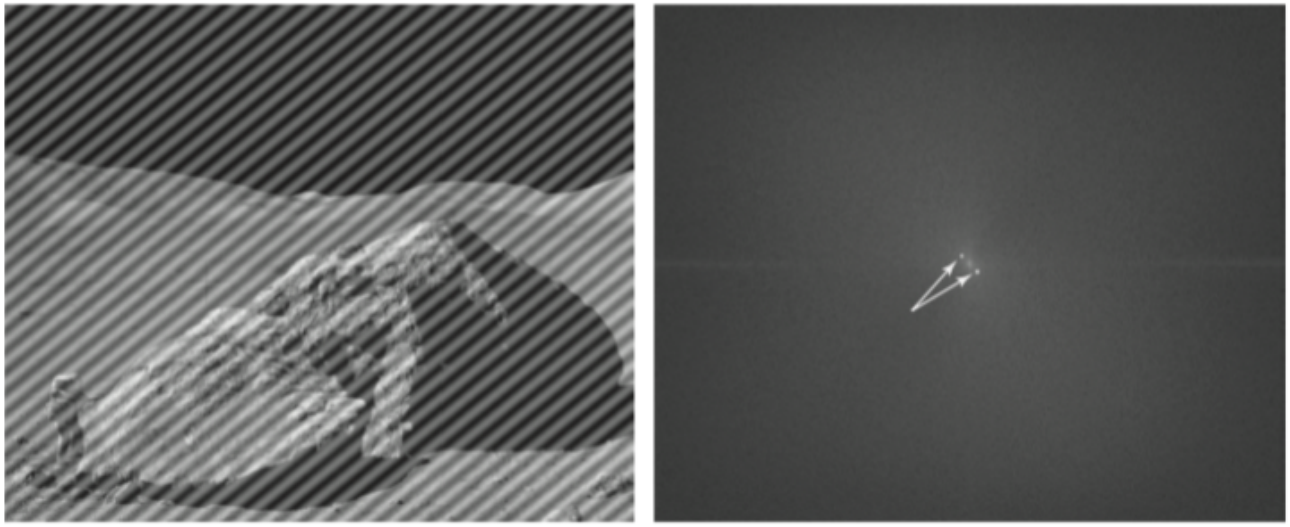
Impulse-type of noises, very special characters

Noise models

Periodic noise

a b

FIGURE 5.5
(a) Image corrupted by additive sinusoidal noise. (b) Spectrum showing two conjugate impulses caused by the sine wave. (Original image courtesy of NASA.)



Remember how we deal with this in the last lecture?
Think about why the FT looks the way it is

Spatial Filtering: Noise Removal

Mean Filters

1. Arithmetic Mean : let $S(x,y) = window(m,n)$

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

2. Geometric Mean :

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

3. Harmonic Mean :

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

Spatial Filtering: Noise Removal

Order Statistics Filters

1. Median Filter : Take a small window $S(x,y)$, find the median and replace (x,y) by this median value.

2. Max filter : $\hat{f}(x,y) = \max_{(s,t)\in S(x,y)} \{g(s,t)\}$ *remove some dark pixels*

3. Min filter : $\hat{f}(x,y) = \min_{(s,t)\in S(x,y)} \{g(s,t)\}$ *remove some white pixels*

4. Mid – point filter : $\hat{f}(x,y) = \frac{1}{2}[Max + Min]$

5. Alpha - trimed Mean Filter : remove all the outliers by excluding $d/2$ brightest and $d/2$ darkest pixels :

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

where $g_r(s,t) = g(s,t)$ excluding $d/2$ brightest and $d/2$ darkest pixels

Spatial Filtering: Noise Removal

6. Adaptive Local Noise Reduction :

$$\hat{f}(x,y) = g(x,y) - \frac{\sigma_N^2}{\sigma_L^2}\left(g(x,y) - m_L\right)$$

m_L = Local mean $= \dfrac{1}{mn} \sum_{(s,t)\in S(x,y)} g(s,t)$

σ_N^2 = Noise variance

σ_L^2 = Local variance

If $\sigma_N^2 << \sigma_L^2$ (less noise) $\Longrightarrow \hat{f} \approx g$

If $\sigma_N^2 \approx \sigma_L^2$ (more noise) $\Longrightarrow$ approx. $\sigma_N^2 \approx \sigma_L^2 \Longrightarrow \hat{f} \approx m_L$

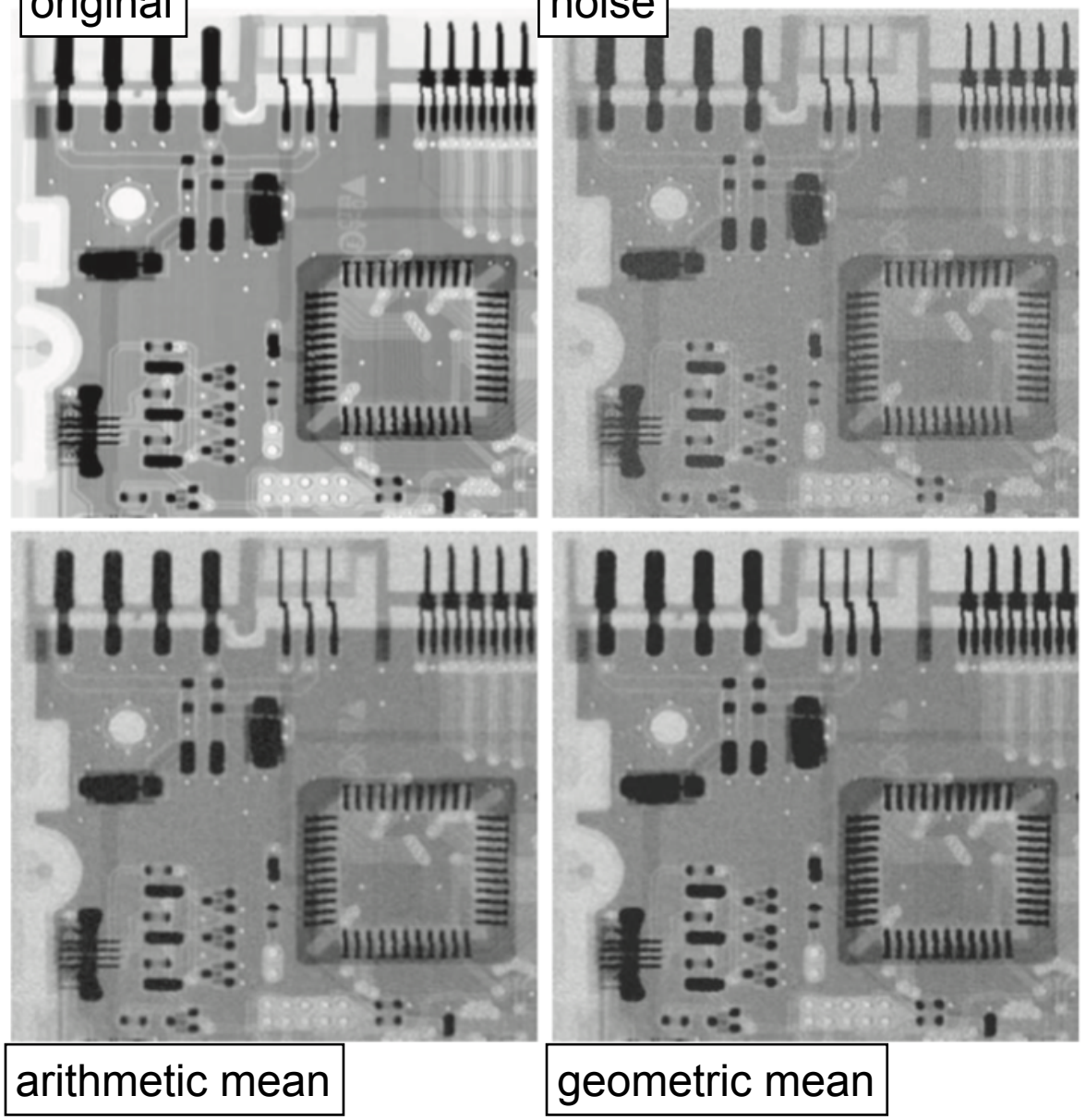
i.e. when there is lot of noise then just average it out the pixels.

For examples on Mean and Order Statistic Filters, see Chapter 5 of G&W Figs. 5.7 to 5.14

Spatial Filtering: Noise Removal

a b
c d

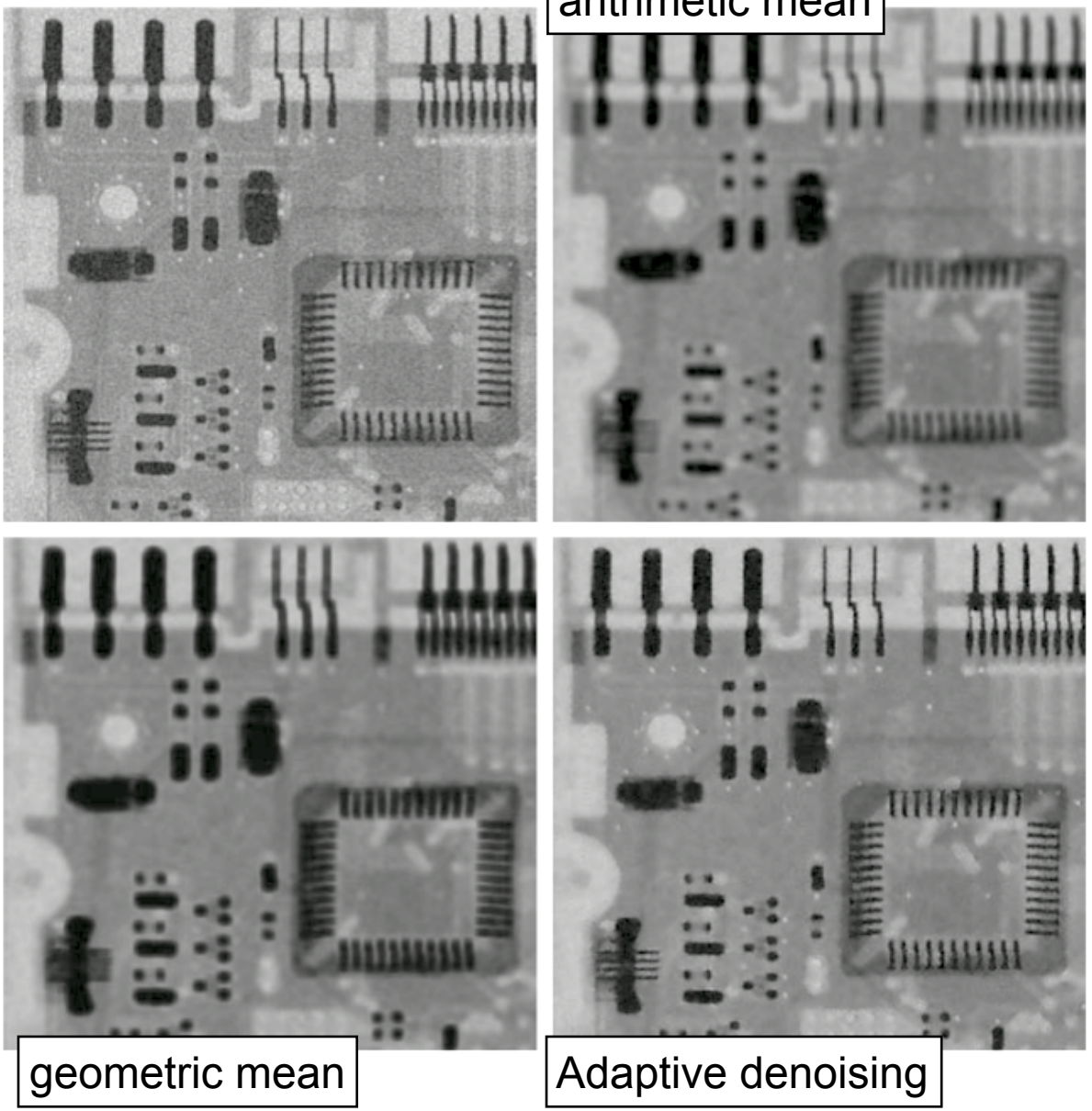
FIGURE 5.7
(a) X-ray image of circuit board. (b) Image corrupted by additive Gaussian noise. (c) Result of filtering with an arithmetic mean filter of size 3×3. (d) Result of filtering with a geometric mean filter of the same size. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)



Spatial Filtering: Noise Removal

a b
c d

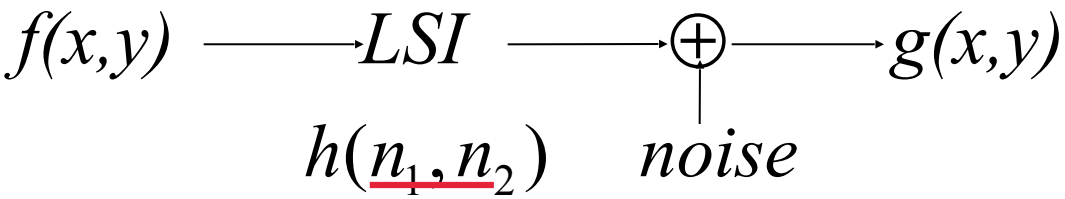
FIGURE 5.13
(a) Image corrupted by additive Gaussian noise of zero mean and a variance of 1000. (b) Result of arithmetic mean filtering. (c) Result of geometric mean filtering. (d) Result of adaptive noise-reduction filtering. All filters used were of size 7 × 7.



Take a break!

Models of Degradation

Degradation can be modeled as:



Use domain specific knowledge to model degradation. For example Study the mechanism that causes the degradation:

Motion Blur Degradation

Consider an analog image *f(x,y)* blurred by a planar motion of the imaging system during exposure.

Let $x_0(t)$ and $y_0(t)$ be the horizontal and vertical translation of *f(x,y)* at the time *t* and *T* is the duration of exposure.

Motion Blur Degradation

$g(x,y)$ = observed image, $f(x,y)$ = original image

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

Take FT both sides :

$$G(\omega_x, \omega_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x,y)\, e^{-j2\pi(\omega_x x + \omega_y y)} dxdy$$

$$= \int_0^T \left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x - x_0(t), y - y_0(t))\, e^{-j2\pi(\omega_x x + \omega_y y)} dxdy \right) dt$$

Let $x - x_0 = X \Longrightarrow x = X + x_0$, similarly for y, hence :

$$G(\omega_x, \omega_y) = \int_0^T \left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(X,Y)\, e^{-j2\pi[\omega_x (X + x_0)]}\, e^{-j2\pi[\omega_y (Y + y_0)]} dXdY \right) dt$$

$$= F(\omega_x, \omega_y) \left[\int_0^T e^{-j2\pi\omega_x x_0}\, e^{-j2\pi\omega_y y_0}\, dt \right] = F(\omega_x, \omega_y)\, H(\omega_x, \omega_y)$$

Motion Blur Degradation

$$G(\omega_x, \omega_y) = F(\omega_x, \omega_y)\, H(\omega_x, \omega_y)$$

Case 1: No motion $\Longrightarrow$ $x_0(t) = y_0(t) = 0$ $\Longrightarrow$ $g(x, y) = f(x, y)$

Case 2: Constant speed in x-direction $\Longrightarrow$ $x_0(t) = \dfrac{at}{T};\ y_0(t) = 0$

Hence: $$H(\omega_x, \omega_y) = \int_0^T e^{-j2\pi\omega_x x_0(t)}\, dt = \int_0^T e^{-j2\pi\omega_x at/T}\, dt$$

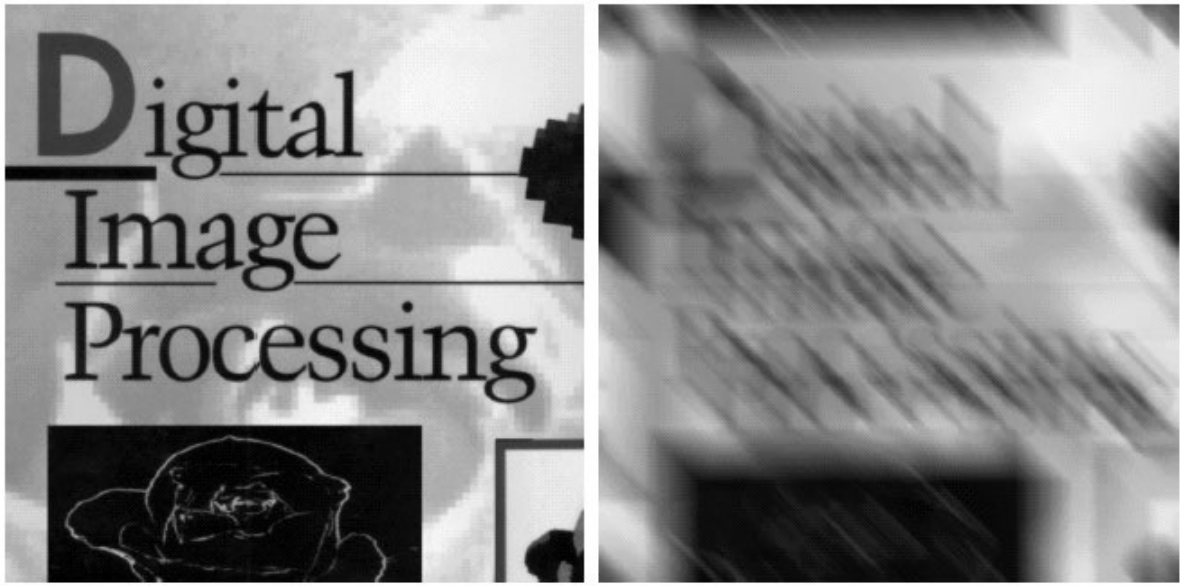
$$H(\omega_x, \omega_y) = \frac{T}{\pi\ \omega_x a} \sin(\pi\ \omega_x a)\, e^{-j\pi\omega_x a}$$

A discrete image $g(n_1, n_2)$ may be approximated by

$$g(n_1, n_2) = f(n_1, n_2) * h(n_1, n_2)$$

Motion Blur Degradation

$$H(\omega_x,\omega_y) = \frac{T}{\pi\,(\omega_x a + \omega_y b)} \sin[\pi\,(\omega_x a + \omega_y b)]\, e^{-j\pi(\omega_x a+\omega_y b)}$$



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 $T = 1$.

Atmospheric Turbulence Degradation

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

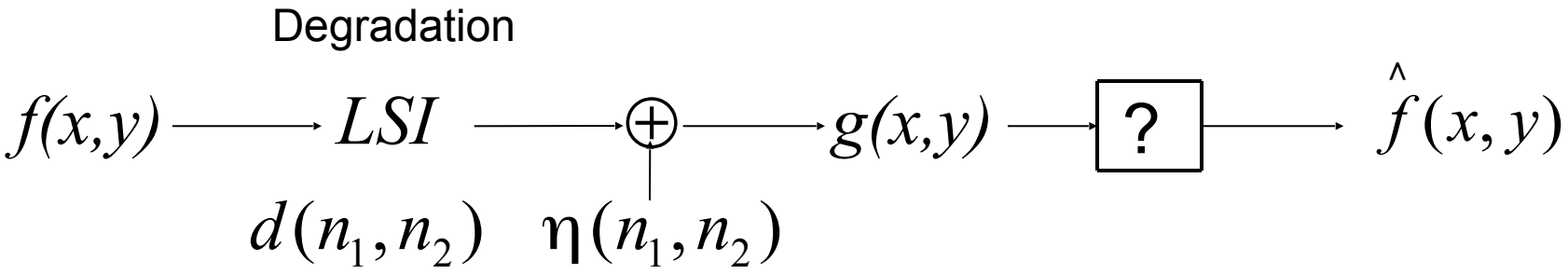
a b
c d

FIGURE 5.25 Modeling turbulence. (a) No visible turbulence. (b) Severe turbulence, $k = 0.0025$. (c) Mild turbulence, $k = 0.001$. (d) Low turbulence, $k = 0.00025$. All images are of size 480×480 pixels. (Original image courtesy of NASA.)

Example:

Look out the window of an airplane, behind the engine, at the ground.

Restoration Problem



Blur (degradation):
- motion
- out-of-focus
- atmospheric turbulence

1. Blur is known: deconvolution
2. Blur is unknown: blind deconvolution

Three classes of Restoration/Deconvolution algorithms:

1. Inverse filtering
2. Least square filtering (e.g. Wiener)
3. Iterative filtering

Inverse Filtering

$$\hat{F}(\omega_1,\omega_2) = \frac{G(\omega_1,\omega_2)}{D(\omega_1,\omega_2)} \Rightarrow H_{inv}(\omega_1,\omega_2) = \frac{1}{D(\omega_1,\omega_2)}$$

But $G(\omega_1,\omega_2) = F(\omega_1,\omega_2)D(\omega_1,\omega_2) + N(\omega_1,\omega_2)$

$$\therefore \quad \hat{F}(\omega_1,\omega_2) = F(\omega_1,\omega_2) + \frac{N(\omega_1,\omega_2)}{D(\omega_1,\omega_2)}$$

Two problems : 1. Noise is not known even when degradation is known

2. $D(\omega_1,\omega_2) \Rightarrow 0$: everything goes bad. In this case limit to the frequencies near the origin to reduce the chance of encountering zero values.

Pseudo-inverse Filter

- Handle zeros in $D(\omega_1, \omega_2)$
  - Treat them separately when performing the inverse filtering:

$$D^{-1}(w_1, w_2) = \begin{cases} \dfrac{1}{D(w_1, w_2)} & |D(w_1, w_2)| > \delta \\ 0 & |D(w_1, w_2)| \leq \delta \end{cases}$$

Pseudo-inverse Filter

- Matlab implementation:

```
x = imread('bean.jpg');
n = 0.2;
figure, imshow(x)
%%
b = fspecial('motion', 50, -30);
B = fft2( b, size(x,1), size(x,2) );

X = fft2(x);
Xb = X.*B;
figure;
imshow(ifft2(Xb),[]);
%%
%%adding noise
xbn = ifft2(Xb) + 10*randn( size(x,1), size(x,2) );
Xbn = fft2(xbn);
figure, imshow( abs( ifft2(Xbn) ), [] );
%%
BF = find( abs(B) < n );
B(BF) = n; %B(BF)=max(max(B))/1.5;

H = ones( size(x,1) , size(x,2) )./B;
Xr = Xbn.*H;

xr = abs( ifft2(Xr) );
figure, imshow( xr, [] );
```

a b
c d

FIGURE 5.27
Restoring Fig. 5.25(b) using Eq. (5-78). (a) Result of using the full filter. (b) Result with H cut off outside a radius of 40. (c) Result with H cut off outside a radius of 70. (d) Result with H cut off outside a radius of 85.

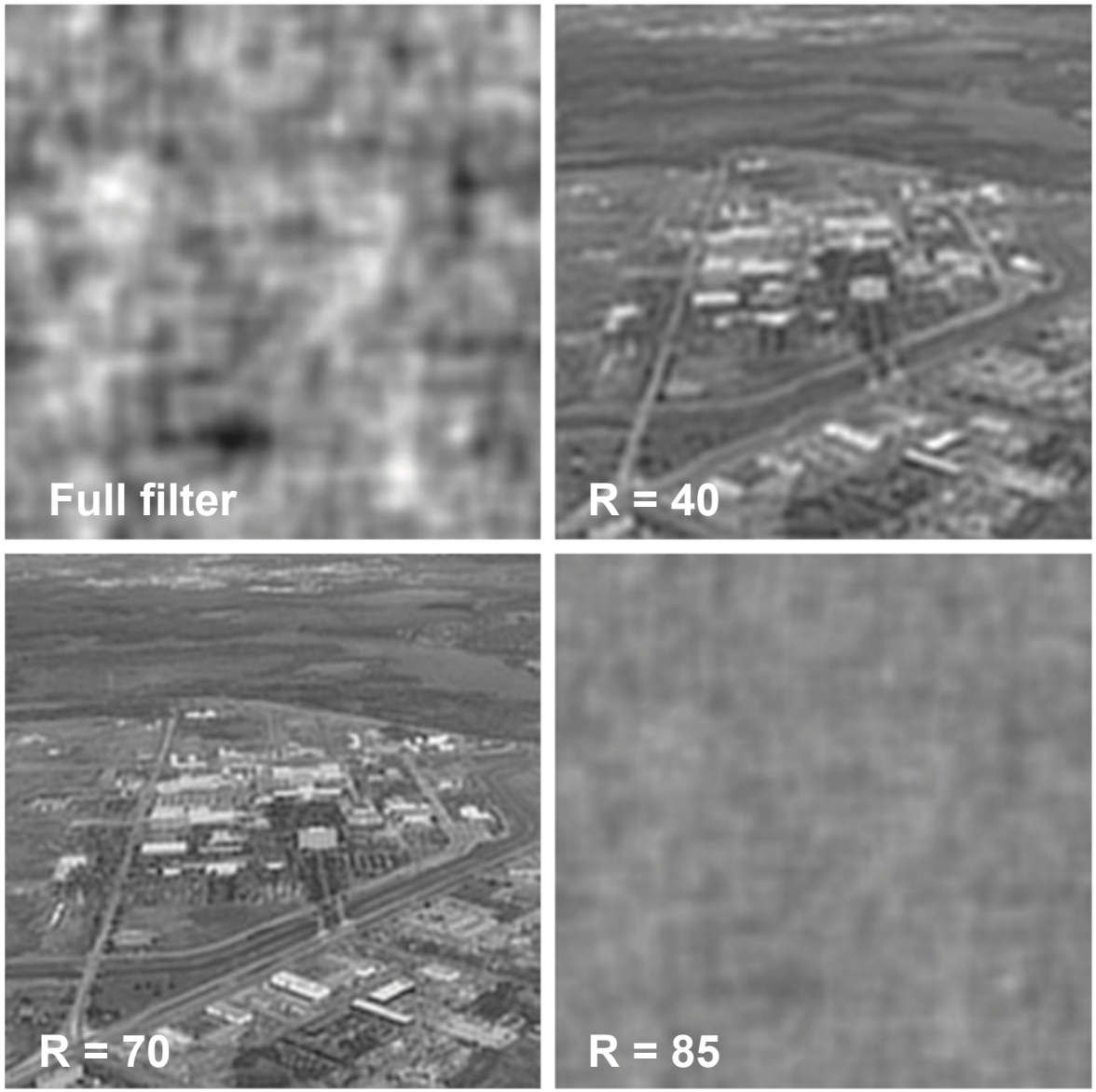
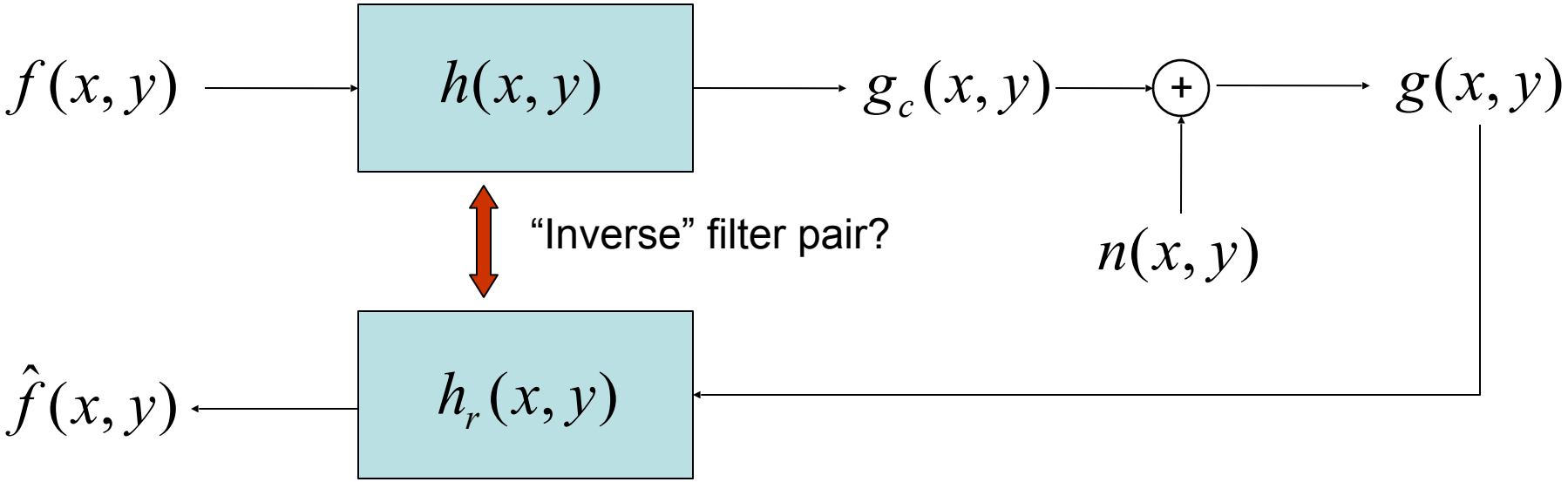


Image Restoration is a Hard Problem



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

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

Wiener Filtering

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

- Offers tradeoff between image recovery and noise suppression.
- Also called Minimum Mean Square Error (MMSE) or Least-Square (LS) filtering

$$e^2 = E\left\{(f - \hat{f})^2\right\}$$

- ✓ *Noise & image are uncorrelated*
- ✓ *One or the other has zero mean*
- ✓ *Recovered image intensity is a linear function of the degraded image*

$$\hat{F}(u,v) = \left[\frac{H^*(u,v)S_f(u,v)}{S_f(u,v)|H(u,v)|^2 + S_\eta(u,v)}\right]G(u,v)$$

$$= \left[\frac{H^*(u,v)}{|H(u,v)|^2 + S_\eta(u,v)/S_f(u,v)}\right]G(u,v)$$

$$= \left[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2 + S_\eta(u,v)/S_f(u,v)}\right]G(u,v)$$

$G(u,v)$

FT of degraded image

$S_\eta(u,v) = |N(u,v)|^2$

power spectrum of noise

$S_f(u,v) = |F(u,v)|^2$

power spectrum of undegraded image

Wiener Filtering

- The power spectrum of the undegraded image is seldom known
- For white noise, the power spectrum is a constant
- The previous equation can be simplified in this case:

$$\hat{F}(u,v) = \left[\frac{1}{H(u,v)} \frac{|H(u,v)|^2}{|H(u,v)|^2 + K} \right] G(u,v)$$

Choose K interactive to achieve the final restoration

ratio of the power spectrum of the noise and signal $S_\eta(u,v)/S_f(u,v)$

$$\text{SNR} = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} |F(u,v)|^2 \Big/ \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} |N(u,v)|^2$$

Wiener Filtering

(noise variance reduced by 2 times)

(noise variance reduced by 5 times)

Inverse filter Wiener filter

a b c
d e f
g h i

FIGURE 5.29 (a) 8-bit image corrupted by motion blur and additive noise. (b) Result of inverse filtering. (c) Result of Wiener filtering. (d)–(f) Same sequence, but with noise variance one order of magnitude less. (g)–(i) Same sequence, but noise variance reduced by five orders of magnitude from (a). Note in (h) how the deblurred image is quite visible through a "curtain" of noise.

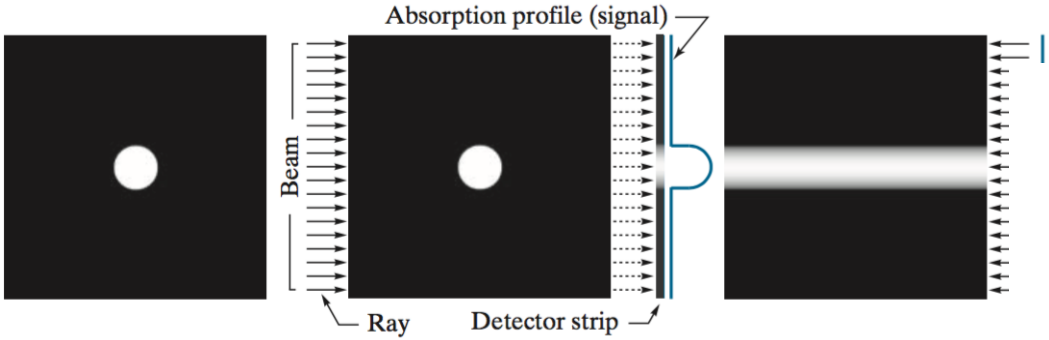
Wiener Filtering

- Matlab implementation:

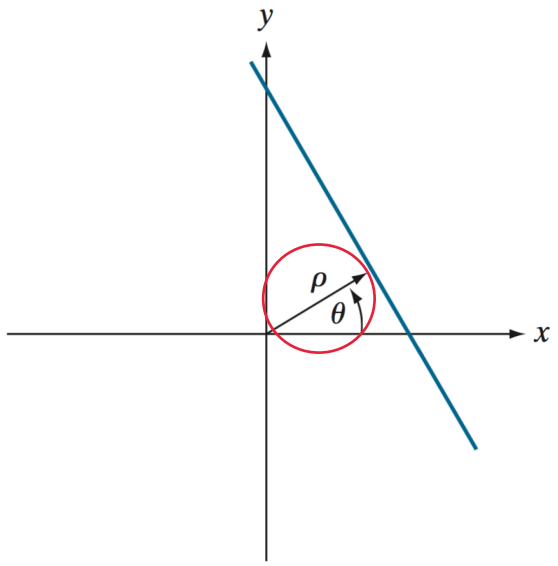
```
I = im2double(imread('bean.jpg'));
figure, imshow(I);
 %%
LEN = 21;
THETA = 11;
PSF = fspecial('motion', LEN, THETA);
blurred = imfilter(I, PSF, 'conv', 'circular');
figure, imshow(blurred);
%%
noise_mean = 0;
noise_var = 0.0001;
blurred_noisy = imnoise(blurred, 'gaussian', noise_mean, noise_var );
figure, imshow(blurred_noisy)
 %%
estimated_nsr = noise_var / var(I(:));
wnr3 = deconvwnr(blurred_noisy, PSF, estimated_nsr);
figure, imshow(wnr3)
```

Reconstruction: Computed Tomography (CT)

- An image in 2-D and a source like X-ray that gives projections on 1-D. How to reconstruct the 2D image?
- Rotate the object (or patient) by an angle that gives another projection.
- $f(x, y)$ is the continuous function in 2-D; and $P(\theta, \rho)$ is the projection of $f(x, y)$ onto the line ρ that forms an angle θ with the *x*-axis.



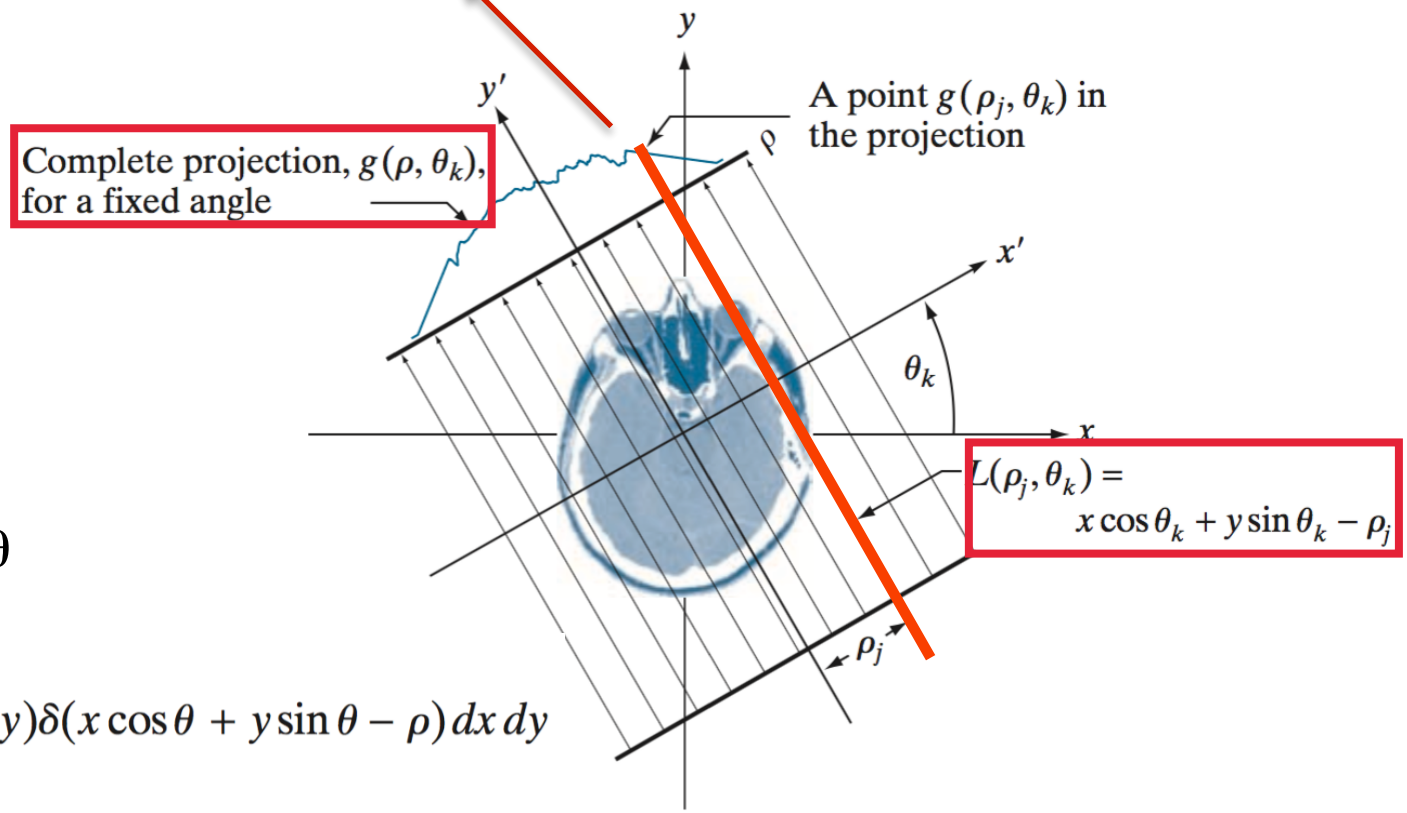
$$x\cos\theta + y\sin\theta = \rho$$



CT: Imaging From Projections

$$g(\rho_j,\theta_k) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x\cos\theta_k + y\sin\theta_k - \rho_j)\,dx\,dy$$

- Projection
- Raysum
- Line Integral



Consider all ρ and θ

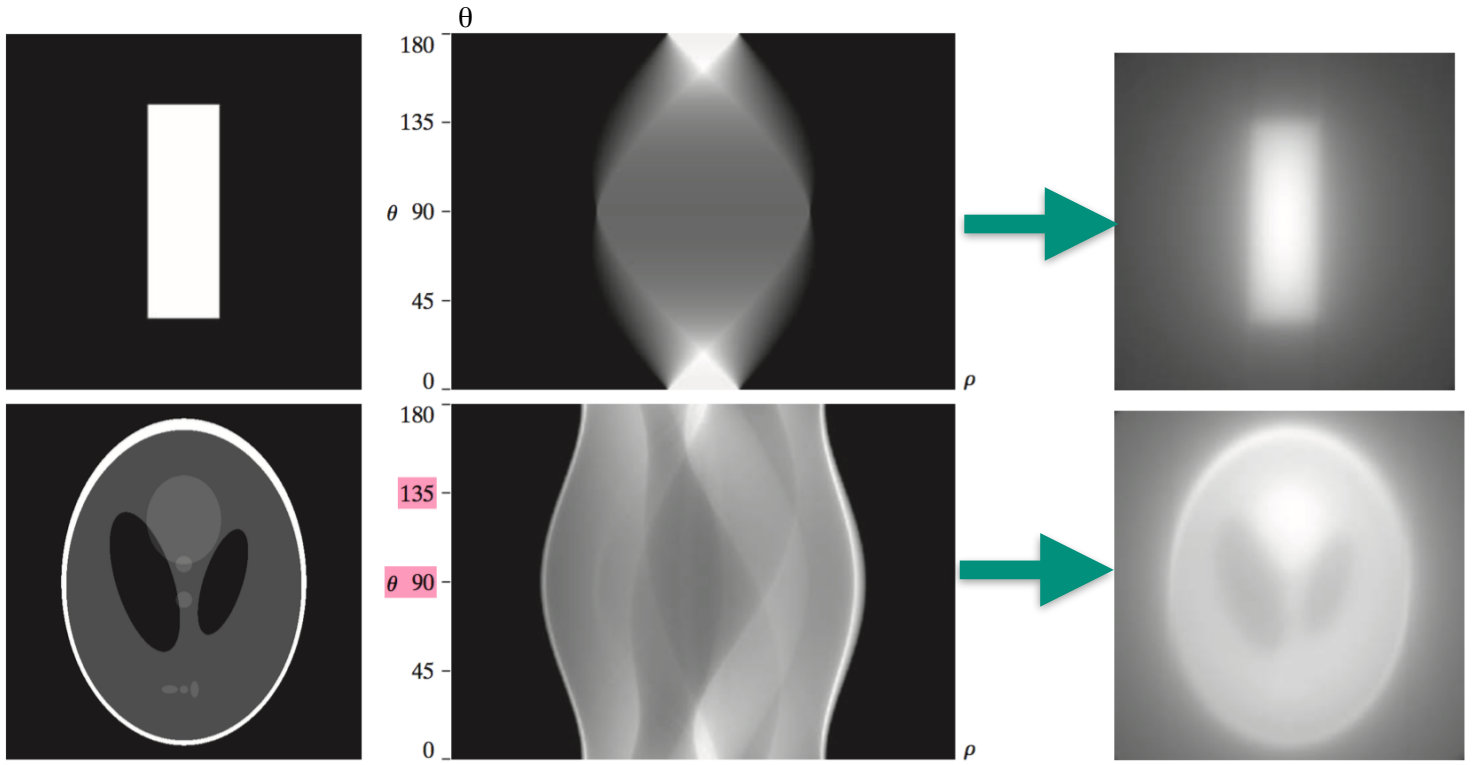
$$g(\rho,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x\cos\theta + y\sin\theta - \rho)\,dx\,dy$$

This equation, which gives the projection (line integral) of *f*(*x*,*y*) along an arbitrary line in the *xy*-plane, is the *Radon transform*

Inverse Problem

Given $P(\theta, \rho)$, find $f(x, y)$. ⟹

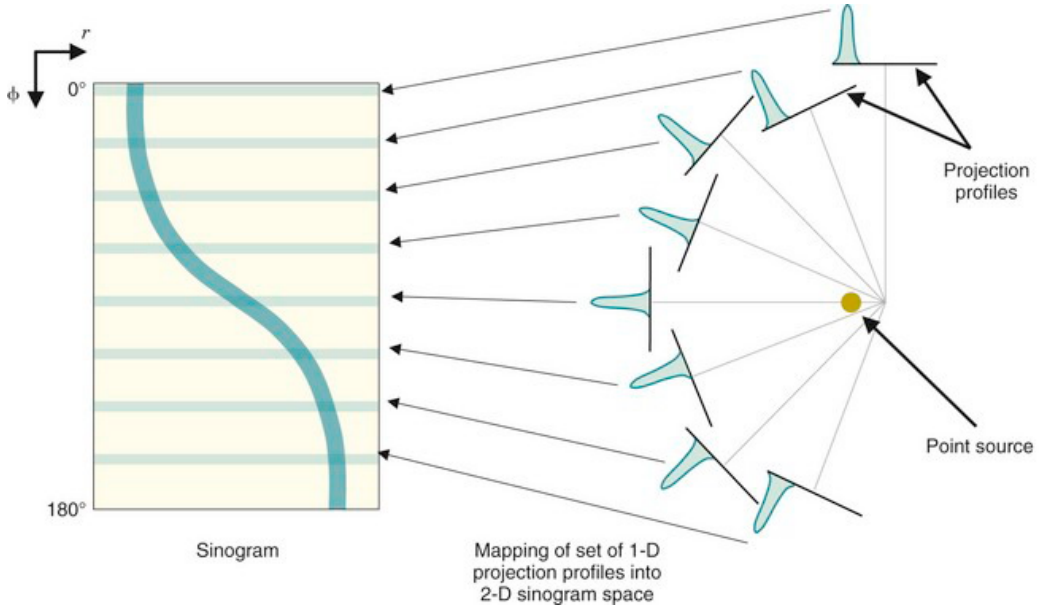
- Incomplete, sparse data
- Noise sensitivity
- Computational complexity

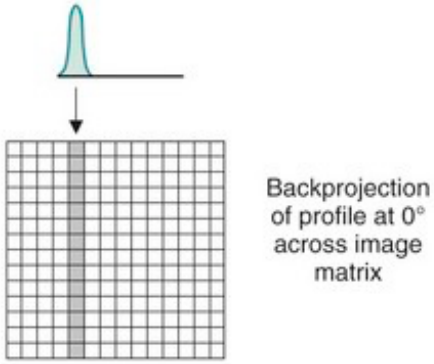


a b
c d

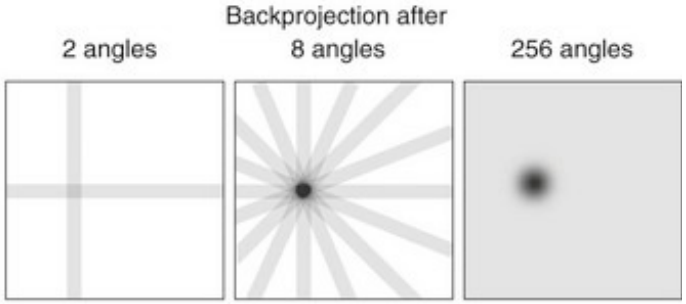
FIGURE 5.39 Two images and their sinograms (Radon transforms). Each row of a sinogram is a projection along the corresponding angle on the vertical axis. (Note that the horizontal axis of the sinograms are values of ρ.) Image (c) is called the *Shepp-Logan phantom*. In its original form, the contrast of the phantom is quite low. It is shown enhanced here to facilitate viewing.

Backprojection





$$f'(x,y) = \frac{1}{N}\sum_{i=1}^{N} p(x\cos\phi_i + y\sin\phi_i, \phi_i)$$



https://radiologykey.com/tomographic-reconstruction-in-nuclear-medicine/

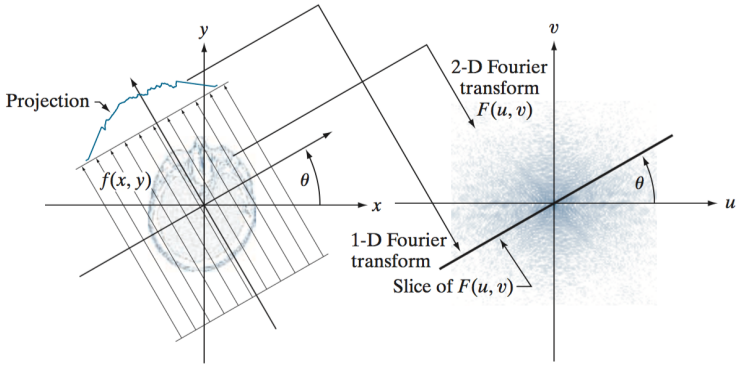
Fourier Domain Reconstruction

The 1-D Fourier transform of a projection with respect to ρ is

$$G(\omega,\theta)=\int_{-\infty}^{\infty} g(\rho,\theta)e^{-j2\pi\omega\rho}\,d\rho$$

$$\begin{aligned}
G(\omega,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x\cos\theta + y\sin\theta - \rho)e^{-j2\pi\omega\rho}dx\,dy\,d\rho \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\left[\int_{-\infty}^{\infty}\delta(x\cos\theta + y\sin\theta - \rho)e^{-j2\pi\omega\rho}d\rho\right]dx\,dy \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-j2\pi\omega(x\cos\theta + y\sin\theta)}dx\,dy
\end{aligned}$$

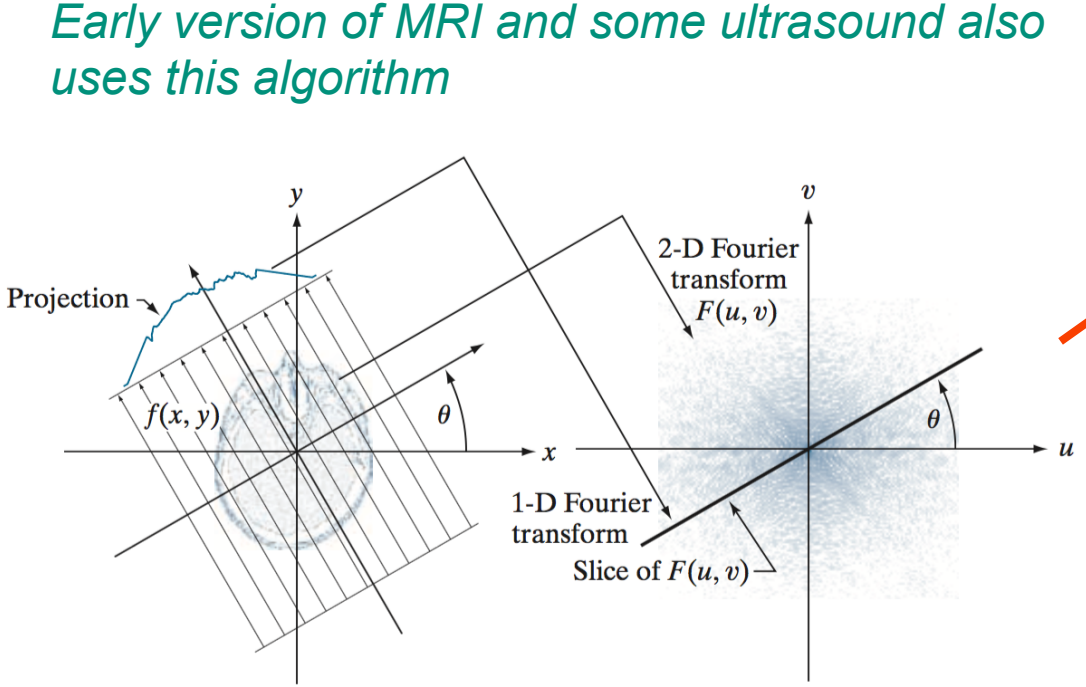
- The FT of the projection (1D) at an angle θ is the slice at that angle of the 2D FT

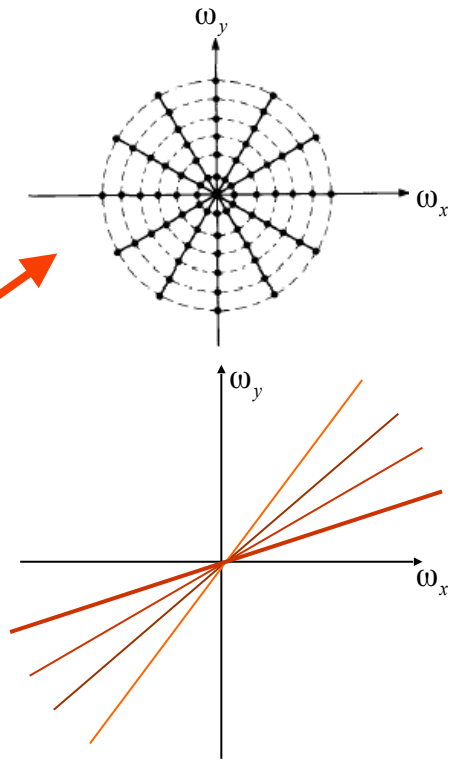


Fourier Domain Reconstruction

1. Collect many projections at diverse angles over the range 0-180 degrees. This yields many central slices in the spectrum
2. Interpolate the given spectral samples from polar to rectangular grid
3. Compute inverse discrete Fourier Transform to obtain an estimate of f(x,y).

Early version of MRI and some ultrasound also uses this algorithm



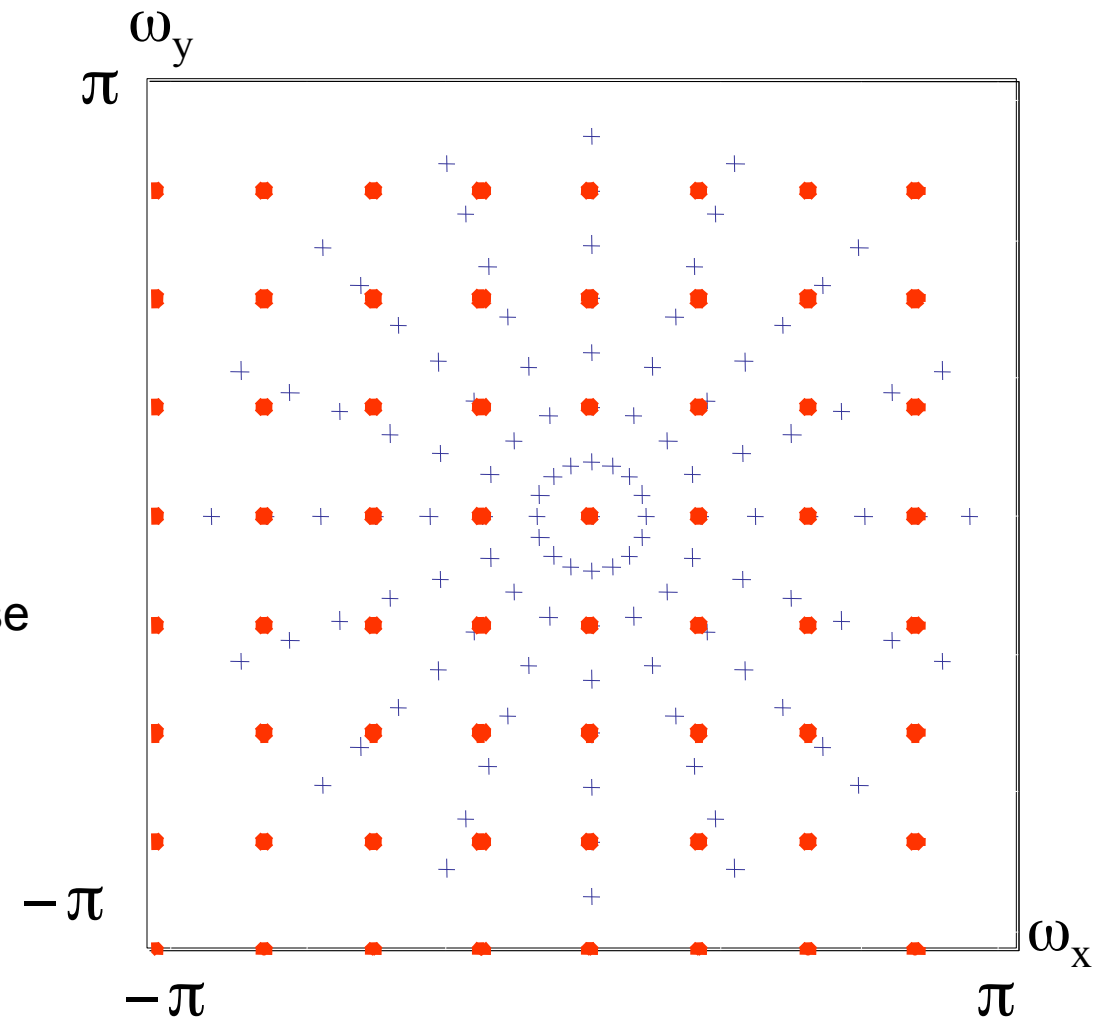


Difficulties of Fourier Reconstruction Method

+ **Polar Samples**

● **Cartesian grid**

Problem: Interpolation is very problematic unless there are very many lines sampled. Sampling is dense near the origin (DC) and very sparse at higher frequencies.



The Projection Slice Theorem

$$G(\omega,\theta) = \left[F(u,v)\right]_{u=\omega\cos\theta;\ v=\omega\sin\theta}$$
$$= F(\omega\cos\theta, \omega\sin\theta)$$

- The FT of the projection (1D) at an angle θ is the slice at that angle of the 2D FT.

Basic idea: Take multiple projections, each projection is a 1D function, then take FT of this projection. This gives a slice of the 2D FT of the object. For many projections at different angles we can construct the 2D FT of the whole object. Inverse 2D FT gives the object $f(x,y)$ itself.

Summary

- How to measure noise level
- Different types of noise, their models, and solutions
- Inverse filter (dealing with noise)
- Wiener fiter
- Backprojection and Radon transformation
- Application: CT image reconstruction
- The projection slice theorem

Reading materials

- Chapter 5
- Page 317-344, 352-363, 368-383

Questions

1. Why simple image blurring is not great for noise reduction? Explain this using the perspective of frequency domain analysis.
2. What are the strategies to handle zero values for the filters in inverse filtering?
3. What are the conditions to ensure that the result of Wiener filtering is successful?
4. What type of filter is the atmospheric turbulence degradation?
5. In what ways does Wiener filtering differ from inverse filtering?
6. How do you control the direction and magnitude of the motion in a motion blur filter?
7. How does local adaptive noise reduction work?
8. What does Radon transform gives us?
9. What is the projection slice theorem?