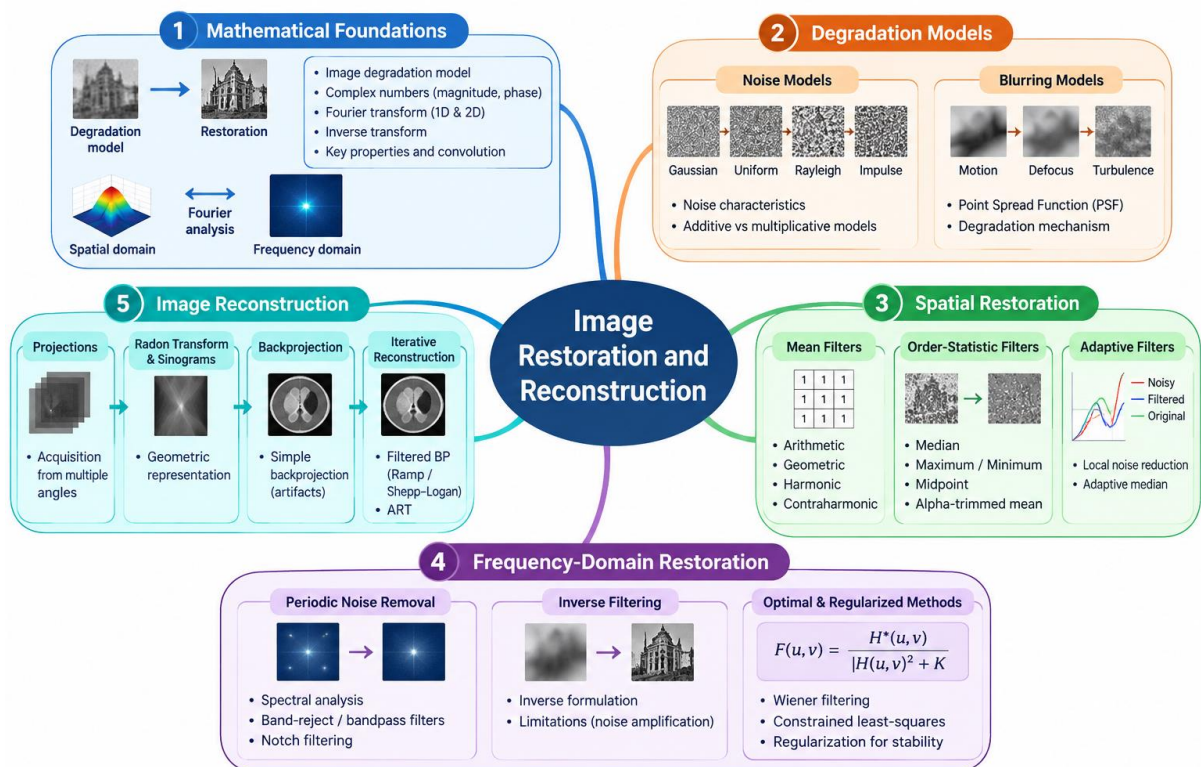


Chapter 4: Image Restoration and Reconstruction

Dr. MAAMAR BEKHOUCHE

m.bekhouche@centre-univ-mila.dz

2026-2027



Overview and Learning Objectives

Real images are never perfect: sensors add noise, lenses are out of focus, cameras move and transmission channels introduce interference. **Image restoration** tries to recover the original image from a degraded version by modelling the degradation and compensating for it. **Image reconstruction** addresses a different problem: the image is not observed directly but must be computed from indirect measurements, the *projections*, as in computed tomography (CT).

The chapter begins with the mathematical tools of frequency analysis (Sections 1 and 2), which are needed both to understand noise and to design filters. It then presents the degradation and noise models (Section 3), the restoration of noisy

images with spatial filters (Section 4) and with frequency-domain filters (Section 5), and finally the reconstruction of images from projections (Section 6).

Learning objectives

After studying this chapter, the student should be able to:

- use complex numbers and the fundamental functions (boxcar, Gaussian, Dirac delta) in signal analysis;
- explain the Fourier series, the continuous Fourier transform and the 2D DFT, and interpret an image spectrum;
- relate convolution in the spatial domain to multiplication in the frequency domain;
- describe the main noise models and estimate their parameters from an image;
- restore noisy images with mean, order-statistic and adaptive filters, and with low-pass, high-pass, band-reject and notch filters;
- explain the principle of computed tomography, the Radon transform and filtered backprojection.

Notation used in this chapter

$f(x, y)$ is the original image, $g(x, y)$ the degraded image, $\hat{f}(x, y)$ the restored estimate, $h(x, y)$ the degradation function and $\eta(x, y)$ the noise. Capital letters (F, G, H, N) denote Fourier transforms. The image size is $M \times N$ pixels and $j = \sqrt{-1}$.

1. Mathematical Tools for Frequency Analysis

1.1. Review of Complex Numbers

The Fourier transform of a signal or of an image is a complex-valued function, so complex numbers are the natural language of frequency analysis. A complex number z is written in rectangular form with a real part a and an imaginary part b , or in polar form with a magnitude $|z|$ and a phase φ (Figure 1), the two forms being linked by Euler's formula:

$$z = a + jb = |z| e^{j\varphi}, \quad e^{j\varphi} = \cos \varphi + j \sin \varphi \quad (1)$$

The **magnitude** $|z|$ and the **phase** φ are obtained from the rectangular components, and the complex conjugate z^* is the mirror image of z with respect to the real axis:

If z is a complex number, then: $a, b \in \mathbb{R}$ et $i^2 = -1$

$$z = a + jb$$

$$= |z| e^{j\varphi}$$

$$= |z|(\cos \theta + i \sin \theta),$$

$$z^* = a - jb = |z| e^{-j\varphi},$$

$$|z| = \sqrt{a^2 + b^2},$$

$$\varphi = \arctan \left(\frac{b}{a} \right),$$

Notations:

- Algebraic: $z = a + jb$
- Cartesian: $z = (a, b)$
- Polar: $z = (r, \theta)$
- Geometric: $z = |z| e^{j\varphi}$
- Trigonometric: $z = |z| (\cos \theta + i \sin \theta)$

$$e^{j\varphi} = \cos \theta + i \sin \theta \quad (2)$$

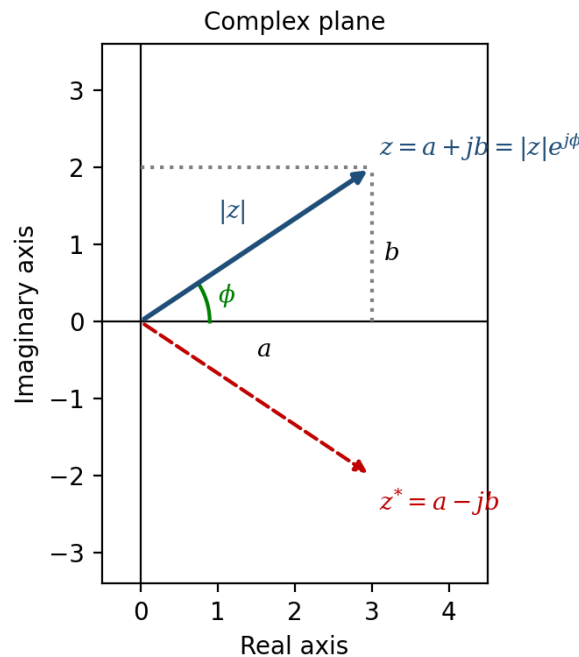


Figure 1: Representation of a complex number and of its conjugate in the complex plane.

Two operations are used constantly. In polar form, the product of two complex numbers multiplies the magnitudes and adds the phases, which is why a shift in the signal domain only changes the *phase* of the Fourier transform:

$$z_1 z_2 = |z_1| |z_2| e^{j(\varphi_1 + \varphi_2)}, \quad z z^* = |z|^2 \quad (3)$$

Finally, a complex exponential is a sinusoid in disguise, because $\cos \varphi = (e^{j\varphi} + e^{-j\varphi})/2$ and $\sin \varphi = (e^{j\varphi} - e^{-j\varphi})/(2j)$. Complex exponentials $e^{j2\pi ux}$ are the elementary building blocks (basis functions) of the Fourier transform.

1.2. Fundamental Functions

Three elementary functions appear throughout the chapter. They are shown in Figure 2 and summarized in Table 1.

1.2.1. Boxcar / Rectangular Function

The boxcar (rectangular) function of width W and height A is equal to A inside the interval $[-W/2, W/2]$ and zero outside:

$$\text{rect}_W(x) = \begin{cases} A & |x| \leq W/2 \\ 0 & |x| > W/2 \end{cases} \quad (4)$$

It models a uniform blur (box filter), an exposure window, an ideal low-pass filter in the frequency domain, or the aperture of a detector.

1.2.2. Gaussian Function

The Gaussian function of standard deviation σ and its 2D version (which is separable and circularly symmetric) are:

$$G(x) = \frac{1}{\sqrt{2\pi} \sigma} e^{-x^2/(2\sigma^2)}, \quad G(x, y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/(2\sigma^2)} \quad (5)$$

It is smooth, has no sharp transition, and is the only function whose Fourier transform is also a Gaussian. It is the standard model for smoothing filters, for blur and for the statistical distribution of noise.

1.2.3. Dirac Delta / Impulse Function

The Dirac delta (unit impulse) is zero everywhere except at the origin and has a unit area:

$$\delta(x) = 0 \quad (x \neq 0), \quad \int_{-\infty}^{\infty} \delta(x) dx = 1 \quad (6)$$

Its fundamental property, the *sifting property*, extracts the value of a function at a point and shows that convolving a function with an impulse reproduces it:

$$\int_{-\infty}^{\infty} f(x) \delta(x - x_0) dx = f(x_0) \quad (7)$$

In the discrete case, the impulse is $\delta(x, y) = 1$ at $x = y = 0$ and 0 elsewhere. It models an ideal point source, and the response of a system to an impulse is its **point spread function**.

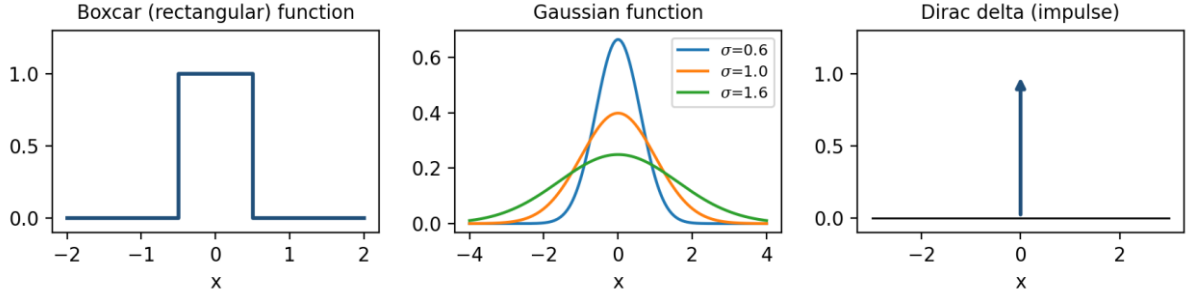


Figure 2: Boxcar function, Gaussian functions of different widths and Dirac impulse.

Table 1: Fundamental functions and their Fourier transforms.

Function	Definition	Fourier transform	Role in image processing
Boxcar	A on $[-W/2, W/2]$, 0 elsewhere	$AW \text{ sinc}(Wu)$	Box blur, ideal low-pass filter, aperture
Gaussian	$e^{-x^2/(2\sigma^2)}/(\sqrt{2\pi}\sigma)$	$e^{-2\pi^2\sigma^2u^2}$ (Gaussian)	Smoothing, blur model, noise model
Dirac delta	$\delta(x)$, unit area at $x = 0$	1 (constant)	Point source, sampling, point spread function

1.3. Concept of Signal Spectrum

The idea of Fourier analysis is that any reasonable signal can be written as a sum of sinusoids of different frequencies, amplitudes and phases. The **spectrum** of the signal is the description of the signal in terms of these ingredients: it shows *which frequencies are present* and *how strongly*. For a signal made of K sinusoids:

$$f(x) = \sum_{k=1}^K A_k \cos(2\pi u_k x + \varphi_k) \quad (8)$$

the amplitude spectrum is a set of lines of height A_k at the frequencies u_k , and the phase spectrum gives φ_k (Figure 3). In an image, the frequency measures how fast the grey level varies in space: **low frequencies** correspond to slowly varying regions (smooth areas, global illumination), and **high frequencies** to rapid variations (edges, fine texture and noise).

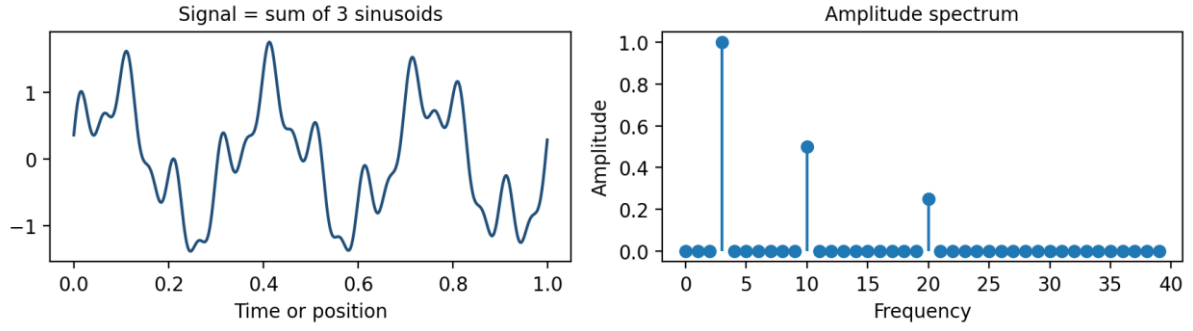


Figure 3: A signal composed of three sinusoids and its amplitude spectrum.

For a complex spectrum $F(u, v) = R(u, v) + jI(u, v)$ we define the magnitude (Fourier spectrum), the phase angle and the power spectrum:

$$|F(u, v)| = \sqrt{R^2(u, v) + I^2(u, v)}, \quad \varphi(u, v) = \arctan \left[\frac{I(u, v)}{R(u, v)} \right] \quad (9)$$

$$P(u, v) = |F(u, v)|^2 = R^2(u, v) + I^2(u, v) \quad (10)$$

The magnitude tells *how much* of each frequency is present, while the phase tells *where* the corresponding components are located; the phase carries most of the structural information of an image.

2. Fourier Analysis and Frequency Domain Processing

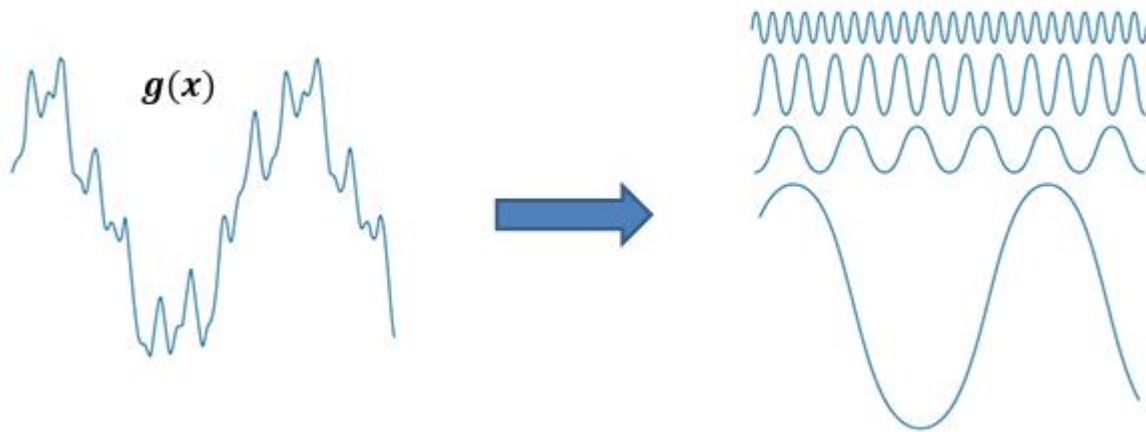
2.1. Fourier Series

2.1.1. Periodic Signals

A periodic signal of period T can be expanded as a sum of harmonically related complex exponentials, the **Fourier series**:

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{j2\pi nt/T}, \quad c_n = \frac{1}{T} \int_{-T/2}^{T/2} f(t) e^{-j2\pi nt/T} dt \quad (11)$$

Consider a periodic signal g with frequency $f_0 = 1/T$. The signal g can be decomposed into a countable (sometimes infinite) sum of sines and cosines. • Example:



The coefficient c_n measures the content of the signal at the frequency n/T (the n -th harmonic). For a real signal, the series can also be written with sines and cosines:

$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left[a_n \cos \frac{2\pi nt}{T} + b_n \sin \frac{2\pi nt}{T} \right] \quad (12)$$

with $a_n = \frac{2}{T} \int f(t) \cos(2\pi nt/T) dt$ and $b_n = \frac{2}{T} \int f(t) \sin(2\pi nt/T) dt$ over one period. For example, a square wave of amplitude ± 1 has only odd harmonics:

$$f(t) = \frac{4}{\pi} \sum_{k=1,3,5,\dots}^{\infty} \frac{1}{k} \sin \frac{2\pi kt}{T} \quad (13)$$

Figure 4 shows how the partial sums converge to the square wave as more harmonics are added; the overshoot near the discontinuities (Gibbs phenomenon) does not disappear. The same phenomenon produces the *ringing* artifacts of ideal filters in Section 5. Finally, the energy of the signal is distributed among the harmonics according to Parseval's relation:

$$\frac{1}{T} \int_{-T/2}^{T/2} |f(t)|^2 dt = \sum_{n=-\infty}^{\infty} |c_n|^2 \quad (14)$$

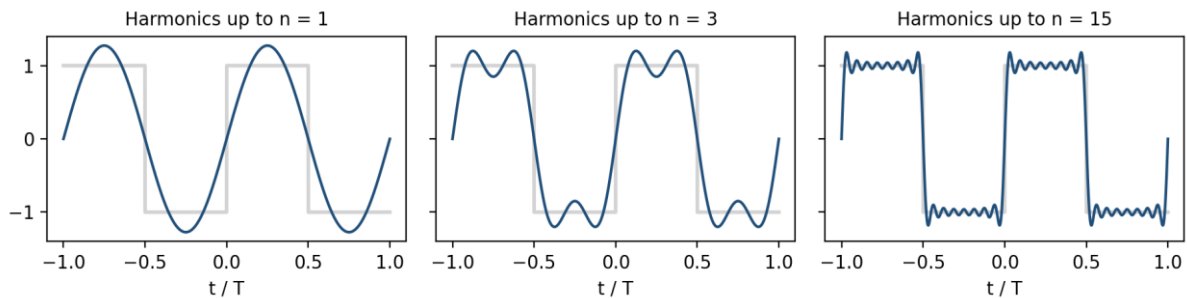


Figure 4: Fourier series approximation of a square wave with an increasing number of harmonics.

2.1.2. Aperiodic Signals

Most signals and images are not periodic. They can be seen as the limit of a periodic signal whose period T tends to infinity. The frequencies n/T of the harmonics then become closer and closer (spacing $1/T \rightarrow 0$) and the discrete sum becomes an integral. Defining $F(u) = \int f(t)e^{-j2\pi ut} dt$ over the whole real line, we have $c_n \approx F(n/T)/T$, and the series becomes:

$$f(t) = \lim_{T \rightarrow \infty} \sum_{n=-\infty}^{\infty} F\left(\frac{n}{T}\right) e^{j2\pi(n/T)t} \frac{1}{T} = \int_{-\infty}^{\infty} F(u) e^{j2\pi ut} du \quad (15)$$

The line spectrum of a periodic signal is replaced by a **continuous spectrum** $F(u)$: this is the Fourier transform of the next section.

2.2. Continuous Fourier Transform (CFT)

2.2.1. 1D Continuous Fourier Transform

The Fourier transform and its inverse form a transform pair:

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} dx \quad (16)$$

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{j2\pi ux} du \quad (17)$$

Two important pairs are $\text{rect}(x) \Leftrightarrow \text{sinc}(u)$ with $\text{sinc}(u) = \sin(\pi u)/(\pi u)$, and the Gaussian pair $e^{-\pi x^2} \Leftrightarrow e^{-\pi u^2}$ (Figure 5). The narrower a function is in the spatial domain, the wider its spectrum: a sharp edge contains many high frequencies, whereas a very smooth function contains few.

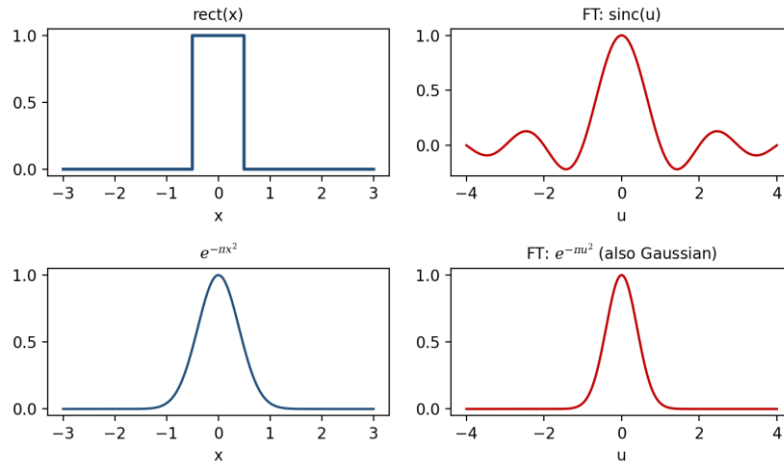


Figure 5: Fourier transform pairs: rectangular function and sinc function, Gaussian function and its Gaussian transform.

2.2.2. 2D Continuous Extension

For an image, the transform is applied along both coordinates, and u and v are the spatial frequencies in the horizontal and vertical directions:

$$F(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi(ux+vy)} dx dy \quad (18)$$

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u, v) e^{j2\pi(ux+vy)} du dv \quad (19)$$

The 2D transform is **separable**: it can be computed as two successive 1D transforms, first along the rows and then along the columns. Each basis function $e^{j2\pi(ux+vy)}$ is a 2D sinusoidal grating whose orientation and spacing are determined by (u, v) .

2.2.3. Properties of the Fourier Transform

The properties of Table 2 explain how operations on the image translate in the frequency domain; they are used all along this chapter.

Table 2: Main properties of the Fourier transform.

Property	Spatial domain	Frequency domain
Linearity	$af_1(x) + bf_2(x)$	$aF_1(u) + bF_2(u)$
Scaling	$f(ax)$	$\frac{1}{ a }F(u/a)$
Shift	$f(x - x_0)$	$F(u) e^{-j2\pi u x_0}$
Modulation	$f(x) e^{j2\pi u_0 x}$	$F(u - u_0)$
Differentiation	df/dx	$j2\pi u F(u)$
Convolution	$f(x) * h(x)$	$F(u) H(u)$
Product	$f(x) h(x)$	$F(u) * H(u)$
Real signal	$f(x)$ real	$F(-u) = F^*(u)$
Parseval	$\int f(x) ^2 dx$	$\int F(u) ^2 du$ (same value)
Rotation (2D)	f rotated by θ_0	F rotated by θ_0

2.3. Discrete Fourier Transform (2D DFT)

2.3.1. Sampling: From Continuous to Discrete Domain

A digital image is a sampled version of a continuous function. Sampling with a step Δx is modelled by multiplying the function by a train of Dirac impulses (a Dirac comb):

$$f_s(x) = f(x) \sum_{n=-\infty}^{\infty} \delta(x - n\Delta x) = \sum_{n=-\infty}^{\infty} f(n\Delta x) \delta(x - n\Delta x) \quad (20)$$

By the properties of Table 2, a product in the spatial domain is a convolution in the frequency domain, and the transform of a Dirac comb is another comb. The spectrum of the sampled function is therefore a periodic repetition (replicas) of the original spectrum:

$$F_s(u) = \frac{1}{\Delta x} \sum_{k=-\infty}^{\infty} F\left(u - \frac{k}{\Delta x}\right) \quad (21)$$

If the function is band-limited, $F(u) = 0$ for $|u| > u_{max}$, the replicas do not overlap as long as the sampling rate satisfies the **Nyquist–Shannon condition**:

$$\frac{1}{\Delta x} \geq 2 u_{max} \quad (22)$$

Otherwise the replicas overlap and high frequencies are folded into low frequencies: this is **aliasing**, which produces false patterns (moiré) that cannot be removed afterwards (Figure 6). In 2D the same condition is applied independently along x and y .

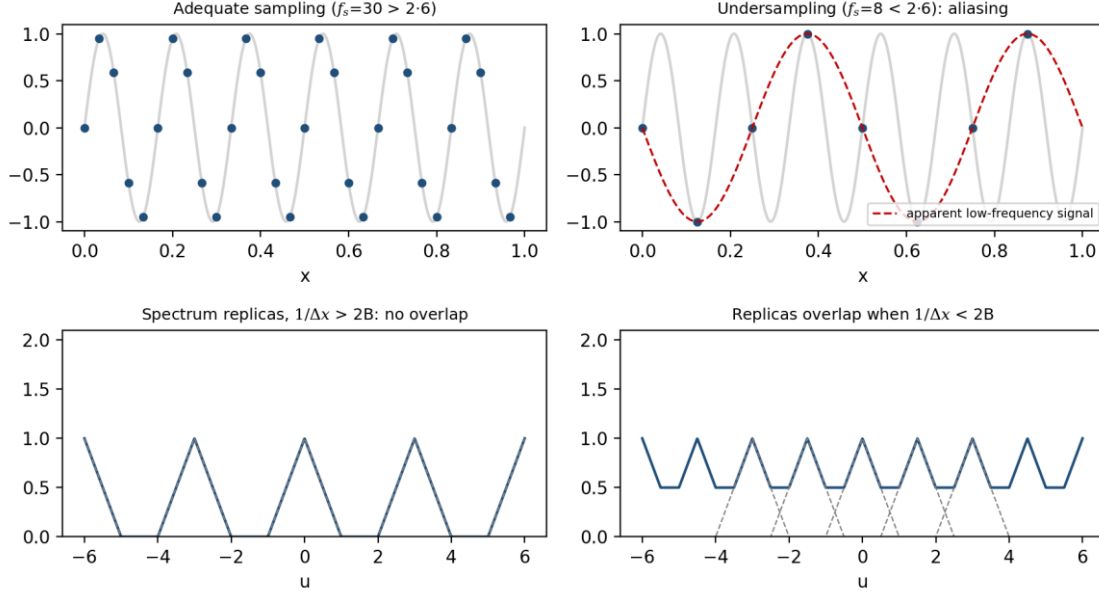


Figure 6: Sampling and aliasing: adequate sampling and undersampling of a sinusoid (top), and the corresponding spectrum replicas (bottom).

2.3.2. Properties and Image Spectrum Representation

For a finite sequence of M samples, only a finite number of frequencies is needed. The **Discrete Fourier Transform (DFT)** and its inverse are:

$$F(u) = \sum_{x=0}^{M-1} f(x) e^{-j2\pi ux/M}, \quad f(x) = \frac{1}{M} \sum_{u=0}^{M-1} F(u) e^{j2\pi ux/M} \quad (23)$$

For an image of size $M \times N$, the 2D DFT and its inverse are:

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)} \quad (24)$$

$$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)} \quad (25)$$

The frequency resolution of the DFT is $\Delta u = 1/(M\Delta x)$: the longer the observed signal, the finer the frequency sampling. The 2D DFT is computed with the Fast Fourier Transform (FFT) algorithm, with a cost proportional to $MN\log(MN)$. Its main properties are the same as those of Table 2, with two specific features: the DFT is **periodic**, $F(u, v) = F(u + M, v) = F(u, v + N)$, and the convolution it

implements is **circular**. In addition, $F(0,0)$ is the sum of all pixel values, which is proportional to the mean grey level.

To display the spectrum, three operations are used. First, the spectrum is **centred** so that the zero frequency is in the middle of the display; this is obtained by multiplying the image by $(-1)^{x+y}$ before the transform, which shifts the spectrum by $(M/2, N/2)$. Second, since $|F(u, v)|$ varies over a huge range (the zero-frequency term dominates), its dynamic range is compressed with a logarithm:

$$s(u, v) = c \log(1 + |F(u, v)|) \quad (26)$$

where c is a scaling constant. Third, the phase is displayed separately if needed (Figure 7). Table 3 gives the main rules for interpreting a centred image spectrum.

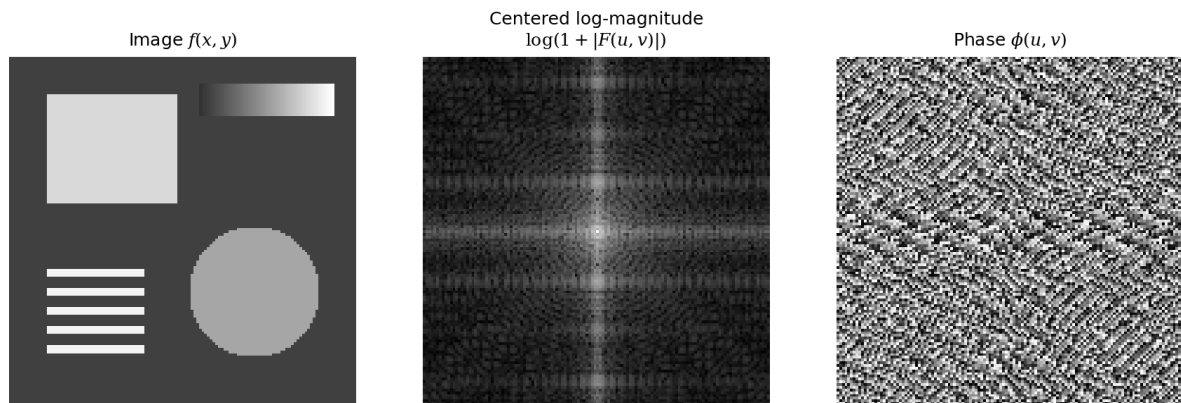


Figure 7: A synthetic image, its centred log-magnitude spectrum and its phase.

Table 3: Interpretation of the features of a centred image spectrum.

Feature of the spectrum	Meaning in the image
Bright centre, energy decreasing outward	Most of the energy is in low frequencies (smooth regions, mean grey level)
Energy spread to large (u, v)	Sharp edges, fine texture or noise
Bright line through the origin	Edges or stripes oriented perpendicular to that line
Isolated bright spikes (symmetric pairs)	Periodic pattern or periodic noise at that frequency and orientation
Rotation of the image	The same rotation of the spectrum

2.4. Example: 2D Discrete Fourier Transform of a 5×5 Image

- Problem Statement

We compute the 2D DFT of the following 5×5 image, which contains a bright 3×3 block (grey level 1) surrounded by a border of zeros (Table 1). Out of the

25 pixels, 9 have $f(x, y) = 1$ and 16 have $f(x, y) = 0$. The index x is the row and y is the column, both starting at 0. The 2D DFT of an $M \times N$ image (with $M = N = 5$ here) is:

$$F(u, v) = \sum_{x=0}^4 \sum_{y=0}^4 f(x, y) e^{-j2\pi(ux+vy)/5} \quad (1)$$

Table 1: The 5×5 image f(x,y).

	y=0	y=1	y=2	y=3	y=4
x=0	0	0	0	0	0
x=1	0	1	1	1	0
x=2	0	1	1	1	0
x=3	0	1	1	1	0
x=4	0	0	0	0	0

Every pixel with $f(x, y) = 0$ contributes nothing to the sum, so only the nine pixels with $f(x, y) = 1$ need to be considered.

- Calculation of Three Pixels of the Spectrum

- Pixel F(0,0)

$$\begin{aligned} F(0, 0) = & f(0, 0) e^{-j2\pi(\frac{0*0}{5} + \frac{0*0}{5})} + f(0, 1) e^{-j2\pi(\frac{0*0}{5} + \frac{1*0}{5})} \\ & + f(0, 2) e^{-j2\pi(\frac{0*0}{5} + \frac{2*0}{5})} + f(0, 3) e^{-j2\pi(\frac{0*0}{5} + \frac{3*0}{5})} \\ & + f(0, 4) e^{-j2\pi(\frac{0*0}{5} + \frac{4*0}{5})} + f(1, 0) e^{-j2\pi(\frac{1*0}{5} + \frac{0*0}{5})} \dots \dots \dots \end{aligned}$$

For $u = v = 0$ all the exponentials are equal to 1. The 16 zero pixels add 0 and the 9 non-zero pixels add 1 each:

$$F(0,0) = \sum_{x=0}^4 \sum_{y=0}^4 f(x, y) = 9 \times 1 + 16 \times 0 = 9$$

Thus $|F(0,0)| = 9$ and the phase is 0° : this value is the sum of all the pixel values.

- Pixel F(1,0)

For $u = 1$ and $v = 0$, the exponential depends only on x . Rows $x = 1, 2, 3$ contain three ones each, while rows $x = 0$ and $x = 4$ contain only zeros and contribute 0:

$$F(1,0) = 3(e^{-j72^\circ} + e^{-j144^\circ} + e^{-j216^\circ})$$

with $e^{-j72^\circ} = 0.309 - 0.951j$, $e^{-j144^\circ} = -0.809 - 0.588j$ and $e^{-j216^\circ} = -0.809 + 0.588j$. The sum in brackets is $-1.309 - 0.951j$, hence:

$$F(1,0) = -3.927 - 2.853j, |F(1,0)| = 4.854, \quad \varphi = -144^\circ \quad (4)$$

- Pixel F(1,1)

Only the nine pixels with $x, y \in \{1,2,3\}$ contribute. Since the exponential factorizes into an x part and a y part, the double sum is the product of two identical sums, each equal to $-1.309 - 0.951j$:

$$\begin{aligned} F(1,1) &= \left(\sum_{x=1}^3 e^{-j2\pi x/5} \right) \left(\sum_{y=1}^3 e^{-j2\pi y/5} \right) \\ &= (-1.309 - 0.951j)^2 = 0.809 + 2.490j \end{aligned} \quad (5)$$

Therefore $|F(1,1)| = 2.618$ and $\varphi = 72^\circ$. The three calculated values are summarized in Table 2.

Table 2: Calculated values of the spectrum.

Pixel	Non-zero terms used	$F(u, v)$	$ F(u, v) $	Phase φ
$F(0,0)$	9 (all with $f = 1$)	9	9	0°
$F(1,0)$	9 (rows 1 to 3)	$-3.927 - 2.853j$	4.854	-144°
$F(1,1)$	9 (block)	$0.809 + 2.490j$	2.618	72°

- Complete Result

The same calculation applied to the 25 frequencies gives the magnitude of the spectrum. It is displayed in Table 3 with the zero frequency at the centre (u, v from -2 to 2). Figure 1 shows the image, the centred magnitude and the logarithmic display $\log(1 + |F(u, v)|)$ used to visualize the spectrum.

Table 3: Magnitude $|F(u,v)|$ of the spectrum (centred).

	u=-2	u=-1	u=0	u=1	u=2
v=-2	0.38	1.00	1.85	1.00	0.38
v=-1	1.00	2.62	4.85	2.62	1.00
v=0	1.85	4.85	9.00	4.85	1.85
v=1	1.00	2.62	4.85	2.62	1.00
v=2	0.38	1.00	1.85	1.00	0.38

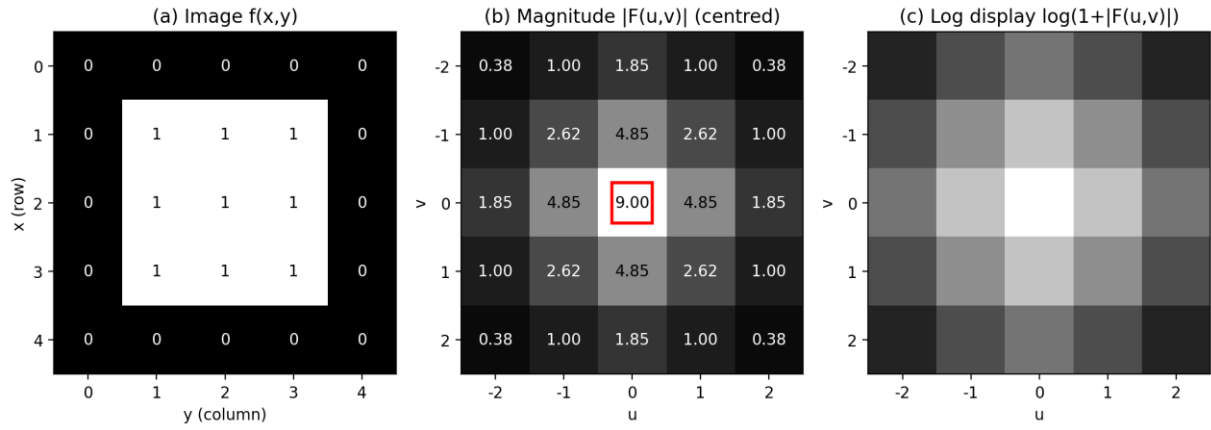


Figure 1: Image $f(x,y)$, centred magnitude $|F(u,v)|$ and logarithmic display of the spectrum (the circled value is $F(0,0) = 9$).

Observations. The energy is concentrated at the centre (low frequencies) and decreases toward the corners; the spectrum is symmetric, as expected for a real image. The largest value, $F(0,0) = 9$, is the sum of all pixel values. The calculated values $F(1,0)$ and $F(1,1)$ appear in Table 3 at the positions $(u, v) = (1,0)$ and $(1,1)$ (shaded).

2.4. Convolution and the Frequency Domain

2.4.1. Spatial Convolution (1D and 2D)

Convolution describes the output of a linear, shift-invariant system (such as a blurring optical system or a spatial filter) with impulse response h . In 1D, for continuous and discrete signals:

$$\begin{aligned} (f * h)(x) &= \int_{-\infty}^{\infty} f(\tau) h(x - \tau) d\tau, & (f * h)(x) \\ &= \sum_m f(m) h(x - m) \end{aligned} \quad (27)$$

and in 2D, for an image f and a filter mask (kernel) h :

$$(f * h)(x, y) = \sum_s \sum_t f(s, t) h(x - s, y - t) \quad (28)$$

The operation consists in *flipping* the kernel (rotating it by 180° in 2D), *sliding* it over the image, and at each position *multiplying* and *summing*. Figure 8 shows the convolution of two rectangular pulses, which gives a triangle: convolution smooths and spreads the signal.

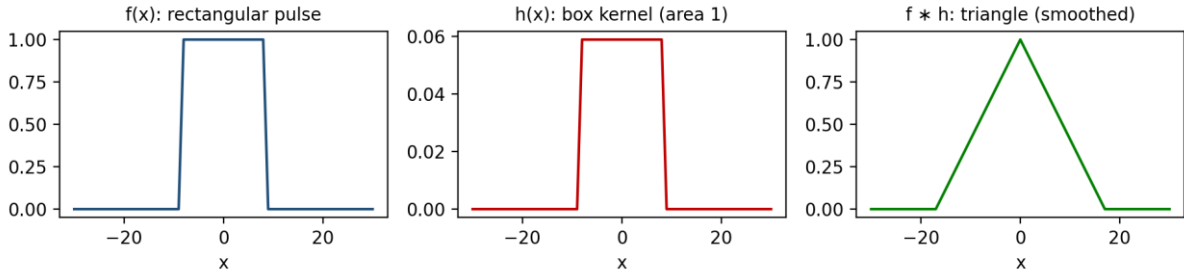


Figure 8: Convolution of a rectangular pulse with a box kernel.

Example – Convolution

1D: for $f = [1, 2, 3, 4]$ and $h = [1, 0, -1]$, the full convolution is $f * h = [1, 2, 2, 2, -3, -4]$. For instance, the third value is $3 \cdot 1 + 2 \cdot 0 + 1 \cdot (-1) = 2$.

2D: for the neighbourhood $[1\ 2\ 3; 4\ 5\ 6; 7\ 8\ 9]$ and the Laplacian mask $[0\ 1\ 0; 1\ -4\ 1; 0\ 1\ 0]$ (symmetric, so flipping has no effect), the output at the centre pixel is $2 + 4 + 6 + 8 - 4 \cdot 5 = 0$: the Laplacian of a linear ramp is zero.

2.4.2. Convolution Theorem and Link to Fourier Domain

The central result for frequency-domain processing is the **convolution theorem**: convolution in the spatial domain is equivalent to a pointwise product in the frequency domain, and conversely:

$$\begin{aligned} f(x, y) * h(x, y) &\Leftrightarrow F(u, v) H(u, v), & f(x, y) h(x, y) \\ &\Leftrightarrow F(u, v) * H(u, v) \end{aligned} \quad (29)$$

The theorem has two consequences. First, a spatial filter of large support can be applied efficiently by FFT, multiplication and inverse FFT. Second, and most important for this chapter, a filter can be *designed directly in the frequency domain*, where its effect on each frequency is explicit: $H(u, v)$ is the **transfer function** of the filter and $h(x, y)$, its inverse transform, is the corresponding spatial mask. Because the DFT is periodic, the product of DFTs corresponds to a *circular* convolution; to obtain the linear convolution, the arrays of sizes $A \times B$ and $C \times D$ must be zero-padded to at least $(A + C - 1) \times (B + D - 1)$ before the transform.

3. Image Degradation and Noise Models

3.1. General Image Degradation Model

The degradation process is modelled as a **degradation function** h acting on the input image $f(x, y)$, followed by the addition of noise $\eta(x, y)$, which produces the degraded image $g(x, y)$. The restoration stage then applies one or several filters to $g(x, y)$ to obtain an estimate $\hat{f}(x, y)$ of the original image (Figure 9).

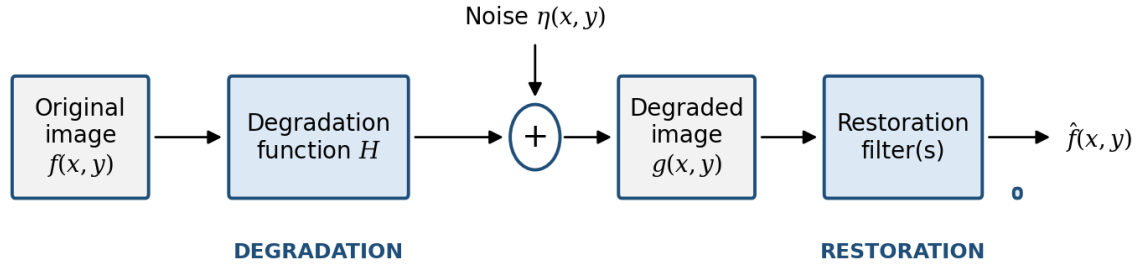


Figure 9: General image degradation and restoration model.

For a linear, position-invariant degradation, the model in the spatial domain and, by the convolution theorem, in the frequency domain, is:

$$g(x, y) = h(x, y) * f(x, y) + \eta(x, y) \quad (30)$$

$$G(u, v) = H(u, v) F(u, v) + N(u, v) \quad (31)$$

The function h is the point spread function of the imaging system (it represents, for example, motion blur, defocus or atmospheric turbulence) and H is its transfer function. Two cases are of particular importance: if there is **only noise** ($H = 1$), then $g = f + \eta$ and the restoration is a denoising problem, treated in Sections 4 and 5; if there is **only blur** ($\eta = 0$), then $G = HF$ and the restoration is a deconvolution problem. In this chapter we concentrate on the removal of noise. Two assumptions are made throughout: the noise is independent of the spatial coordinates (except for periodic noise), and it is uncorrelated with the image.

3.2. Noise Typology and Mathematical Models

Noise can be classified according to three criteria. According to its **spatial behaviour**, it is either *random* (white) noise, whose value at a pixel is independent of the neighbouring pixels, or *periodic* noise, which has a regular spatial structure. According to its **relation to the image**, it is *additive* or *multiplicative*. According to its **statistical distribution**, it is described by a probability density function (PDF) $p(z)$ of the grey-level values z . Figure 11 shows the visual appearance of the main models.

3.2.1. Random Noise Models

Gaussian noise arises from electronic circuit noise and sensor noise under poor illumination or high temperature. It has the PDF:

$$p(z) = \frac{1}{\sqrt{2\pi} \sigma} e^{-(z-\mu)^2/(2\sigma^2)} \quad (32)$$

with mean μ and variance σ^2 . **Uniform noise** assumes that all the values between a and b are equally likely:

$$p(z) = \begin{cases} \frac{1}{b-a} & a \leq z \leq b \\ 0 & \text{otherwise} \end{cases} \quad (33)$$

Rayleigh noise has a skewed PDF and is used in range imaging:

$$p(z) = \begin{cases} \frac{2}{b} (z-a) e^{-(z-a)^2/b} & z \geq a \\ 0 & z < a \end{cases} \quad (34)$$

Erlang (gamma) noise and its special case, **exponential noise** ($b = 1$), appear in laser imaging:

$$p(z) = \begin{cases} \frac{a^b z^{b-1}}{(b-1)!} e^{-az} & z \geq 0 \\ 0 & z < 0 \end{cases} \quad p(z) = a e^{-az} \quad (z \geq 0) \quad (35)$$

Impulse (salt-and-pepper) noise appears as isolated black and white pixels, caused by faulty switching, transmission errors or dead sensor elements:

$$p(z) = \begin{cases} P_a & z = a \\ P_b & z = b \\ 0 & \text{otherwise} \end{cases} \quad (36)$$

If P_a and P_b are both non-zero and roughly equal, the noise looks like salt (grey level b) and pepper (grey level a) grains. The PDFs of four of these models are plotted in Figure 10 and their main parameters are given in Table 4.

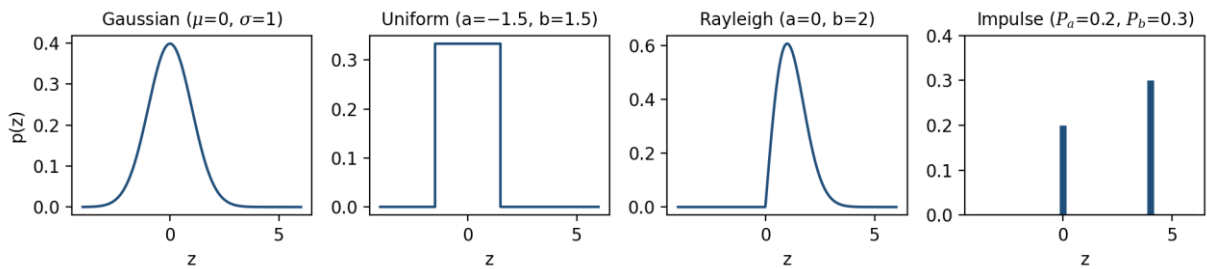


Figure 10: Probability density functions of the Gaussian, uniform, Rayleigh and impulse noise models.

Table 4: Summary of the random noise models.

Model	Mean	Variance	Typical origin
Gaussian	μ	σ^2	Sensor and electronic noise
Uniform	$(a + b)/2$	$(b - a)^2/12$	Quantization, random number generators
Rayleigh	$a + \sqrt{\pi b/4}$	$b(4 - \pi)/4$	Range imaging
Erlang	b/a	b/a^2	Laser imaging
Exponential	$1/a$	$1/a^2$	Laser imaging
Impulse	Depends on P_a, P_b, a, b	Depends on P_a, P_b, a, b	Faulty sensors, transmission errors

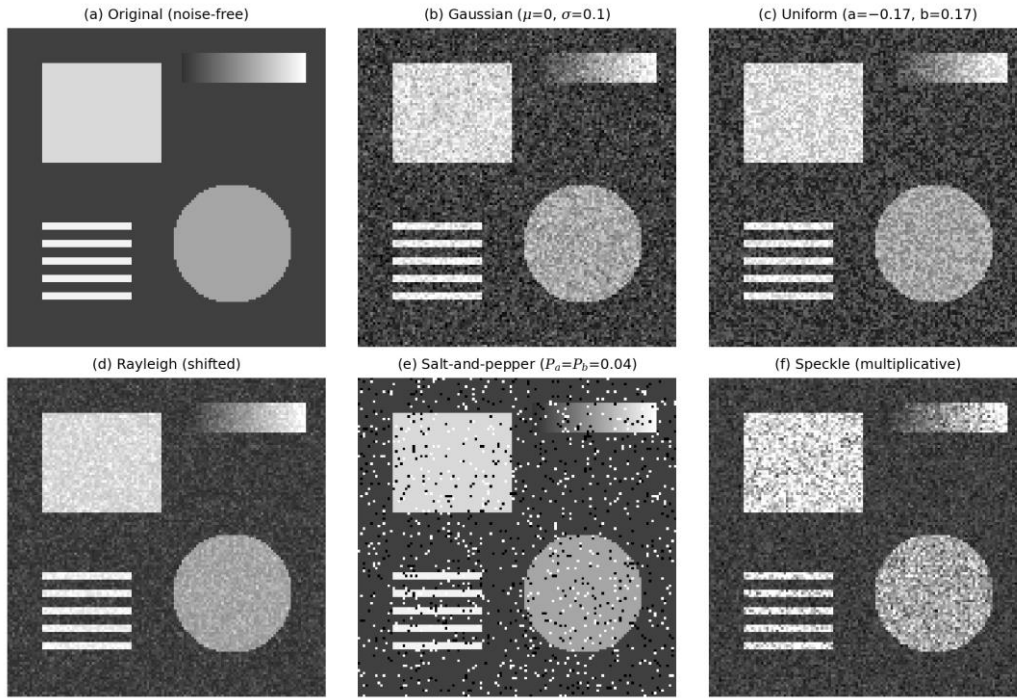


Figure 11: Appearance of the main noise models added to a synthetic image (simulation).

3.2.2. Additive versus Multiplicative Noise and Speckle

In the **additive** model, the noise is added to the ideal image and its amplitude does not depend on the image content. In the **multiplicative** model, the noise amplitude is proportional to the local intensity:

$$\begin{aligned} g(x, y) &= f(x, y) + \eta(x, y) & (\text{additive}), \\ &= f(x, y) \eta(x, y) & (\text{multiplicative}) \end{aligned} \quad (37)$$

The most important example of multiplicative noise is **speckle**, which affects ultrasound, radar (SAR) and laser images and is caused by the random interference of coherent waves scattered by the target. Bright regions are noisier

than dark ones. A multiplicative model can be converted into an additive one with a logarithm, $\ln g = \ln f + \ln \eta$, so that additive-noise filters can be used before taking the exponential.

3.2.3. Periodic Noise

Periodic noise is caused by electrical or electromechanical interference during acquisition. It is spatially dependent and appears as regular patterns such as stripes. A sinusoidal interference of amplitude A is described by:

$$\eta(x, y) = A \sin \left(2\pi \left(\frac{u_0 x}{M} + \frac{v_0 y}{N} \right) \right) \quad (38)$$

Its Fourier transform is a pair of impulses at $(\pm u_0, \pm v_0)$, which makes it easy to remove in the frequency domain (Section 5.3).

3.3. Noise Parameter Estimation

The parameters of a noise model are usually unknown and must be estimated from the image or from the imaging system. When the sensor is available, images of a uniform scene (for example a grey card) can be acquired. Otherwise, a small region of interest (ROI) with an approximately **constant grey level** is selected in the image (Figure 12): in such a region, the variations of the grey levels are due to the noise only. From the histogram of the ROI, the normalized probability $p(z_i)$ of each grey level z_i is computed, and the mean and variance are estimated by:

$$\bar{z} = \sum_{i=0}^{L-1} z_i p(z_i), \quad \sigma^2 = \sum_{i=0}^{L-1} (z_i - \bar{z})^2 p(z_i) \quad (39)$$

The shape of the histogram indicates the type of noise (bell-shaped for Gaussian, flat for uniform, skewed for Rayleigh, two isolated peaks for impulse noise). The parameters of the model are then deduced from $\bar{z}$ and σ^2 by inverting the formulas of Table 4:

$$\begin{array}{ll} \text{Gaussian:} & \mu = \bar{z}, \sigma^2 \\ & b = \bar{z} + \sqrt{3\sigma^2} \end{array} \quad \text{Uniform:} \quad a = \bar{z} - \sqrt{3\sigma^2}, \quad (40)$$

$$\begin{aligned}
&\text{Rayleigh: } b = \frac{4\sigma^2}{4 - \pi}, \\
&a = \bar{z} - \sqrt{\frac{\pi b}{4}} \quad \text{Erlang: } a = \frac{\bar{z}}{\sigma^2}, \\
&b = \frac{\bar{z}^2}{\sigma^2}
\end{aligned} \tag{41}$$

For **impulse noise**, the probabilities P_a and P_b are directly read as the heights of the two spikes of the histogram. For **periodic noise**, the frequencies and amplitudes of the interference are estimated from the peaks of the Fourier spectrum. The procedure is summarized below:

1. Select a region of interest with a constant grey level (flat background).
2. Compute its histogram and normalize it to obtain $p(z_i)$.
3. Identify the shape of the histogram to choose the noise model.
4. Estimate $\bar{z}$ and σ^2 with the equation above and deduce the parameters of the model.

Noisy image with a flat region (ROI)

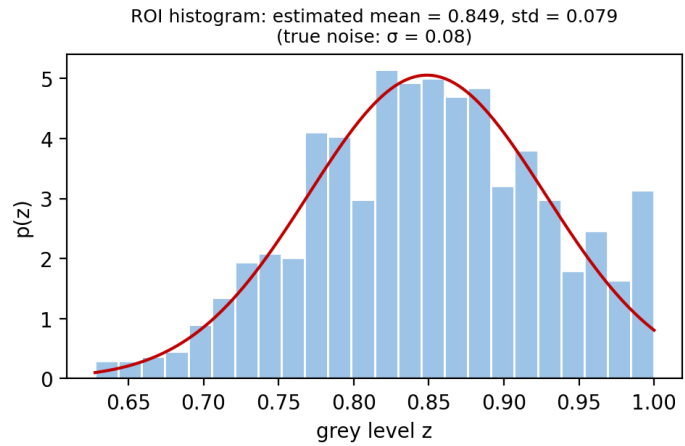
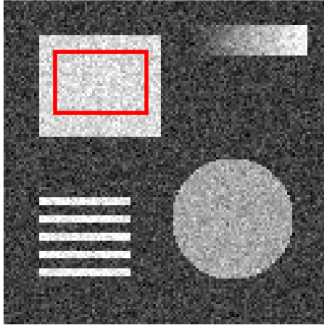


Figure 12: Estimation of Gaussian noise parameters from a flat region of interest (simulation).

4. Restoration of Noisy Images using Spatial Filters

When the only degradation is noise, $g(x, y) = f(x, y) + \eta(x, y)$, the restoration can be performed directly in the spatial domain by sliding a window over the image. Let S_{xy} denote the set of coordinates of a rectangular window of size $m \times n$ centred at (x, y) . The restored value $\hat{f}(x, y)$ is computed from the grey levels of the degraded image inside S_{xy} .

4.1. Mean-Based Filters (Arithmetic, Geometric, Harmonic Mean, etc.)

The **arithmetic mean filter** replaces each pixel by the average of its neighbourhood. It smooths local variations and reduces Gaussian and uniform noise, at the price of blurring the edges:

$$\hat{f}(x, y) = \frac{1}{mn} \sum_{(s,t) \in S_{xy}} g(s, t) \quad (42)$$

The **geometric mean filter** gives a smoothing comparable to the arithmetic mean but loses less image detail. It fails if any pixel of the window is zero:

$$\hat{f}(x, y) = \left[\prod_{(s,t) \in S_{xy}} g(s, t) \right]^{1/mn} \quad (43)$$

The **harmonic mean filter** works well for salt noise and Gaussian noise, but fails for pepper noise (a zero value makes the denominator infinite):

$$\hat{f}(x, y) = \frac{mn}{\sum_{(s,t) \in S_{xy}} \frac{1}{g(s, t)}} \quad (44)$$

The **contraharmonic mean filter** has an order parameter Q :

$$\hat{f}(x, y) = \frac{\sum_{(s,t) \in S_{xy}} g(s, t)^{Q+1}}{\sum_{(s,t) \in S_{xy}} g(s, t)^Q} \quad (45)$$

For $Q > 0$ the filter eliminates *pepper* noise; for $Q < 0$ it eliminates *salt* noise; it cannot do both at the same time. For $Q = 0$ it reduces to the arithmetic mean and for $Q = -1$ to the harmonic mean. Using the wrong sign of Q amplifies the noise instead of removing it. The behaviour of the four filters is summarized in Table 5.

Table 5: Comparison of the mean-based restoration filters.

Filter	Main effect	Suited for	Limitation
Arithmetic mean	Smooths local variations	Gaussian, uniform noise	Blurs edges and details
Geometric mean	Smoothing with less detail loss	Gaussian noise	Output is 0 if one pixel of the window is 0
Harmonic mean	Attenuates bright outliers	Salt noise, Gaussian noise	Fails with pepper noise
Contra-harmonic mean	Eliminates one type of impulse	Salt ($Q < 0$) or pepper ($Q > 0$) noise	Wrong sign of Q makes the result worse

4.2. Order-Statistic Filters (Median, Min/Max, Midpoint, etc.)

Order-statistic filters are non-linear filters based on *ranking* the grey levels inside the window and selecting one of them. The **median filter** replaces the pixel by the median of its neighbourhood:

$$\hat{f}(x, y) = \text{median}_{(s,t) \in S_{xy}} \{g(s, t)\} \quad (46)$$

It is very effective against impulse noise and, for a comparable window size, blurs much less than the arithmetic mean (Figure 13). The **maximum** and **minimum** filters select the brightest and the darkest value, respectively:

$$\hat{f}(x, y) = \max_{(s,t) \in S_{xy}} \{g(s, t)\}, \quad \hat{f}(x, y) = \min_{(s,t) \in S_{xy}} \{g(s, t)\} \quad (47)$$

The maximum filter removes pepper noise (but brightens the image) and the minimum filter removes salt noise (but darkens it). The **midpoint filter** averages the two extremes and works best for randomly distributed noise:

$$\hat{f}(x, y) = \frac{1}{2} \left[\max_{(s,t) \in S_{xy}} \{g(s, t)\} + \min_{(s,t) \in S_{xy}} \{g(s, t)\} \right] \quad (48)$$

The **alpha-trimmed mean filter** deletes the $d/2$ lowest and the $d/2$ highest grey levels and averages the remaining $mn - d$ pixels $g_r(s, t)$:

$$\hat{f}(x, y) = \frac{1}{mn - d} \sum_{(s,t) \in S_{xy}} g_r(s, t), \quad 0 \leq d \leq mn - 1 \quad (49)$$

For $d = 0$ the filter is the arithmetic mean, and for $d = mn - 1$ it becomes the median filter. Intermediate values are a good compromise when impulse and Gaussian noise are present together.

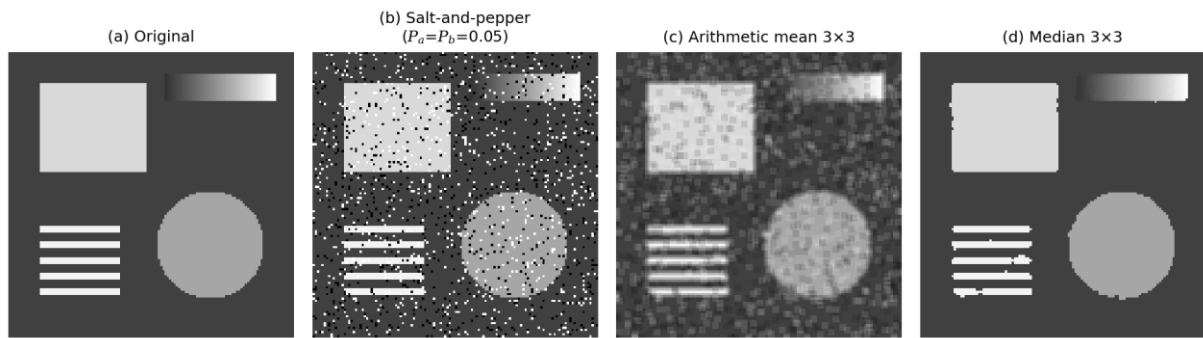


Figure 13: Effect of the arithmetic mean and median filters on a synthetic image corrupted by salt-and-pepper noise (simulation).

Example – Output of each filter for one window

Consider the following 3×3 window around a pixel corrupted by salt noise (255). Sorted, the values are 9, 10, 10, 10, 11, 11, 12, 12, 255.

Window		
10	12	11
9	255	10
11	12	10

Results: arithmetic mean 37.8; geometric mean 15.1; harmonic mean 11.8; contraharmonic mean 239.4 for $Q = 1.5$ and 10.7 for $Q = -1.5$; median 11; maximum 255; minimum 9; midpoint 132; alpha-trimmed mean ($d = 4$) 10.8. The outlier strongly corrupts the arithmetic mean, the midpoint and the contraharmonic filter with $Q > 0$, whereas the median, the harmonic mean, the alpha-trimmed mean and the contraharmonic filter with $Q < 0$ ignore it.

Table 6: Summary of the order-statistic filters.

Filter	Output	Best suited for	Remark
Median	Middle value of the sorted window	Salt-and-pepper noise	Preserves edges better than the mean
Maximum	Largest grey level	Pepper noise	Brightens the image
Minimum	Smallest grey level	Salt noise	Darkens the image
Midpoint	$[\max + \min]/2$	Gaussian, uniform noise	Sensitive to impulses
Alpha-trimmed mean	Mean after removing $d/2$ extremes on each side	Mixed impulse and Gaussian noise	d controls the trade-off

4.3Example: DFT, Noise Filtering and Inverse DFT of a 2×2 mage

1. Problem Statement

Consider the following 2×2 image, where x is the row index and y the column index, both starting at 0:

Table 1: The input image $i(x,y)$.

	y=0	y=1
x=0	2	1
x=1	5	3

The following three operations are applied step by step: (1) the 2D Discrete Fourier Transform (DFT), (2) a noise filter in the frequency domain, and (3) the Inverse DFT (IDFT) of the filtered spectrum.

2. Step 1: Two-Dimensional DFT

For an image of size $M \times N$ with $M = N = 2$, the 2D DFT is:

$$\begin{aligned}
 F(u, v) &= \sum_{x=0}^1 \sum_{y=0}^1 i(x, y) e^{-j2\pi(ux+vy)/2} \\
 &= \sum_{x=0}^1 \sum_{y=0}^1 i(x, y) (-1)^{ux+vy}
 \end{aligned} \tag{1}$$

because $e^{-j\pi k} = (-1)^k$ for an integer k . The spectrum has four values, which are computed one by one. For each of them, the four pixels are multiplied by the factor $(-1)^{ux+vy}$ and added.

2.1. Pixel F(0,0)

For $u = v = 0$ all the factors are +1:

$$F(0,0) = 2(+1) + 1(+1) + 5(+1) + 3(+1) = 11 \quad (2)$$

2.2. Pixel F(0,1)

For $u = 0$ and $v = 1$, the factor is $(-1)^y$: it is -1 for the pixels of the second column:

$$F(0,1) = 2(+1) + 1(-1) + 5(+1) + 3(-1) = 3 \quad (3)$$

2.3. Pixel F(1,0)

For $u = 1$ and $v = 0$, the factor is $(-1)^x$: it is -1 for the pixels of the second row:

$$F(1,0) = 2(+1) + 1(+1) + 5(-1) + 3(-1) = -5 \quad (4)$$

2.4. Pixel F(1,1)

For $u = v = 1$, the factor is $(-1)^{x+y}$ (a checkerboard pattern):

$$F(1,1) = 2(+1) + 1(-1) + 5(-1) + 3(+1) = -1 \quad (5)$$

All the values are real, so the phase is 0° for the positive values and 180° for the negative ones. Table 2 gives the result.

Table 2: Spectrum F(u,v) of the image.

(u, v)	$F(u, v)$	Magnitude $ F(u, v) $	Phase φ $\varphi = \arctan(b/a)$	Meaning
(0,0)	11	11	0°	Sum of the pixels (mean $\times 4$)
(0,1)	3	3	0°	Horizontal variation (columns)
(1,0)	-5	5	180°	Vertical variation (rows)
(1,1)	-1	1	180°	Diagonal (checkerboard) variation, highest frequency

Pour $F(1,0) = -5$, le nombre complexe est réel et négatif : sa partie réelle vaut $a = -5$ et sa partie imaginaire vaut $b = 0$.

Calcul

Avec $\varphi = \arctan(b/a) = \arctan(0/-5) = \arctan(0)$, on obtient 0° ou 180° . Cette formule seule ne distingue pas les deux cas, car $\arctan$ ne tient pas compte du signe de a . Il faut utiliser la fonction à deux arguments **$\arctan2(b, a)$** , qui prend en compte les signes de a et b :

$\varphi = \arctan2(0, -5) = 180^\circ$ (c'est-à-dire π rad)

Vérification : avec la forme polaire $F = |F| \cdot e^{j\varphi} = 5 \cdot e^{j\pi} = 5 \cdot (\cos 180^\circ + j \sin 180^\circ) = 5 \cdot (-1 + 0j) = -5$ ✓

Règle pratique pour un nombre réel :

- positif (par exemple $F(0,1) = 3$) : $\varphi = 0^\circ$;
- négatif (par exemple $F(1,0) = -5$) : $\varphi = 180^\circ$.

Cela correspond à ce qui est indiqué dans le tableau du document : la phase vaut 180° pour $F(1,0)$ et pour $F(1,1) = -1$.

In matrix form, with the rows indexed by u and the columns by v :

	$v=0$	$v=1$
$u=0$	11	3
$u=1$	-5	-1

3. Step 2: Noise Filtering in the Frequency Domain

Noise, like edges and fine details, is mostly made of **high spatial frequencies**. A simple noise filter is therefore a **low-pass filter**, which multiplies each spectrum value by a transfer function $H(u, v)$ equal to 1 for the frequencies to keep and 0 for those to reject:

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

$$H(u, v) = \begin{cases} 1 & D(u, v) \leq D_0 \\ 0 & D(u, v) > D_0 \end{cases}, \quad D(u, v) = \sqrt{u^2 + v^2} \quad (6)$$

For a 2×2 image the only frequencies are 0 and 1 in each direction. We choose the cutoff **$D_0 = 1$** : the components (0,0), (0,1) and (1,0), with $D = 0, 1$ and 1 , are kept, and the component (1,1), which has $D = \sqrt{2} > D_0$ and is the highest-frequency (checkerboard) component, is rejected. (The frequencies are used in their DFT order, with the zero frequency at the corner, so no centring is needed.)

Table 3: Filter $H(u,v)$ and filtered spectrum $G(u,v) = H(u,v) F(u,v)$.

(u, v)	$D(u, v)$	$H(u, v)$	$F(u, v)$	Calculation	$G(u, v)$
(0,0)	0	1	11	1×11	11
(0,1)	1	1	3	1×3	3
(1,0)	1	1	-5	$1 \times (-5)$	-5
(1,1)	$\sqrt{2} \approx 1.41$	0	-1	$0 \times (-1)$	0

The filtered spectrum is:

	v=0	v=1
u=0	11	3
u=1	-5	0

4. Step 3: Inverse Two-Dimensional DFT

The image is reconstructed from the filtered spectrum with the inverse DFT:

$$\begin{aligned}
 i'(x, y) &= \frac{1}{MN} \sum_{u=0}^1 \sum_{v=0}^1 G(u, v) e^{j2\pi(ux+vy)/2} \\
 &= \frac{1}{4} \sum_{u=0}^1 \sum_{v=0}^1 G(u, v) (-1)^{ux+vy}
 \end{aligned} \tag{7}$$

where $i'(x, y)$ is the filtered image. The factor $(-1)^{ux+vy}$ is now attached to the *output pixel* (x, y) . Each pixel is computed in turn, with $G(0,0) = 11$, $G(0,1) = 3$, $G(1,0) = -5$ and $G(1,1) = 0$.

4.1. Pixel $i'(0,0)$

$$\begin{aligned}
 i'(0,0) &= \frac{1}{4} [11(+1) + 3(+1) + (-5)(+1) + 0(+1)] = \frac{9}{4} \\
 &= 2.25
 \end{aligned} \tag{8}$$

4.2. Pixel $i'(0,1)$

Here $x = 0$ and $y = 1$, so the factor is $(-1)^v$:

$$\begin{aligned}
 i'(0,1) &= \frac{1}{4} [11(+1) + 3(-1) + (-5)(+1) + 0(-1)] = \frac{3}{4} \\
 &= 0.75
 \end{aligned} \tag{9}$$

4.3. Pixel $i'(1,0)$

Here $x = 1$ and $y = 0$, so the factor is $(-1)^u$:

$$i'(1,0) = \frac{1}{4}[11(+1) + 3(+1) + (-5)(-1) + 0(-1)] = \frac{19}{4} = 4.75 \quad (10)$$

4.4. Pixel $i'(1,1)$

Here $x = y = 1$, so the factor is $(-1)^{u+v}$:

$$i'(1,1) = \frac{1}{4}[11(+1) + 3(-1) + (-5)(-1) + 0(+1)] = \frac{13}{4} = 3.25 \quad (11)$$

5. Result

The filtered image is:

Table 4: Filtered image $i'(x,y)$ obtained after the inverse DFT.

	y=0	y=1
x=0	2.25	0.75
x=1	4.75	3.25

Table 5 compares it with the original image.

Table 5: Original image and filtered image.

Pixel (x,y)	Original $i(x,y)$	Filtered $i'(x,y)$	Difference $i' - i$
(0,0)	2	2.25	+0.25
(0,1)	1	0.75	-0.25
(1,0)	5	4.75	-0.25
(1,1)	3	3.25	+0.25

Checks and remarks

- **Mean preserved:** the sum of the pixels is 11 before and after the filter ($2.25 + 0.75 + 4.75 + 3.25 = 11$), because the component $F(0,0)$, which carries the mean, has been kept.
- **Perfect reconstruction without filtering:** applying the IDFT to the unfiltered spectrum gives back the original image. For example, $i(0,0) = \frac{1}{4}[11 + 3 - 5 - 1] = 2$ and $i(1,0) = \frac{1}{4}[11 + 3 + 5 + 1] = 5$.
- **Effect of the filter:** the rejected component $F(1,1) = -1$ contributed $\frac{1}{4}(-1)(-1)^{x+y} = -0.25(-1)^{x+y}$ to each pixel. Removing it therefore adds $+0.25$ to the pixels $(0,0)$ and $(1,1)$ and -0.25 to the pixels $(0,1)$ and $(1,0)$, which is exactly the difference column of Table 5. With only four pixels, the change is small and is not a visible smoothing.

4.3. Adaptive Filters

The previous filters use the same rule everywhere in the image. **Adaptive filters** change their behaviour according to the statistics of the pixels under the window, which gives better results at a higher computational cost.

4.3.1. Adaptive Local Noise-Reduction Filter

This filter uses the noisy value $g(x, y)$, the noise variance σ_η^2 (known or estimated as in Section 3.3), the local mean m_L and the local variance σ_L^2 computed in S_{xy} :

$$\hat{f}(x, y) = g(x, y) - \frac{\sigma_\eta^2}{\sigma_L^2} [g(x, y) - m_L] \quad (50)$$

The behaviour depends on the ratio σ_η^2/σ_L^2 :

- if $\sigma_\eta^2 = 0$, the filter returns $g(x, y)$ unchanged (no noise to remove);
- if $\sigma_L^2 \gg \sigma_\eta^2$ (high local contrast, typically an edge), the filter returns a value close to $g(x, y)$ and the edge is preserved;
- if $\sigma_L^2 \approx \sigma_\eta^2$ (the local variance is due to the noise), the filter returns the arithmetic mean m_L ;
- if $\sigma_L^2 < \sigma_\eta^2$ (possible because the noise variance is only an estimate), the ratio is set to 1 so that $\hat{f} = m_L$.

4.3.2. Adaptive Median Filter

The adaptive median filter uses a window whose size increases up to a maximum size S_{max} . In the window S_{xy} , let z_{min} , z_{max} and z_{med} be the minimum, maximum and median grey levels and z_{xy} the grey level at (x, y) . The algorithm has two stages (Figure 14):

1. **Stage A:** compute $A_1 = z_{med} - z_{min}$ and $A_2 = z_{med} - z_{max}$. If $A_1 > 0$ and $A_2 < 0$, the median is not an impulse: go to Stage B. Otherwise, increase the window size; if the size is still $\leq S_{max}$, repeat Stage A, else output z_{xy} .
2. **Stage B:** compute $B_1 = z_{xy} - z_{min}$ and $B_2 = z_{xy} - z_{max}$. If $B_1 > 0$ and $B_2 < 0$, the pixel itself is not an impulse and the output is z_{xy} (the original value, which preserves details). Otherwise, the pixel is an impulse and the output is z_{med} .

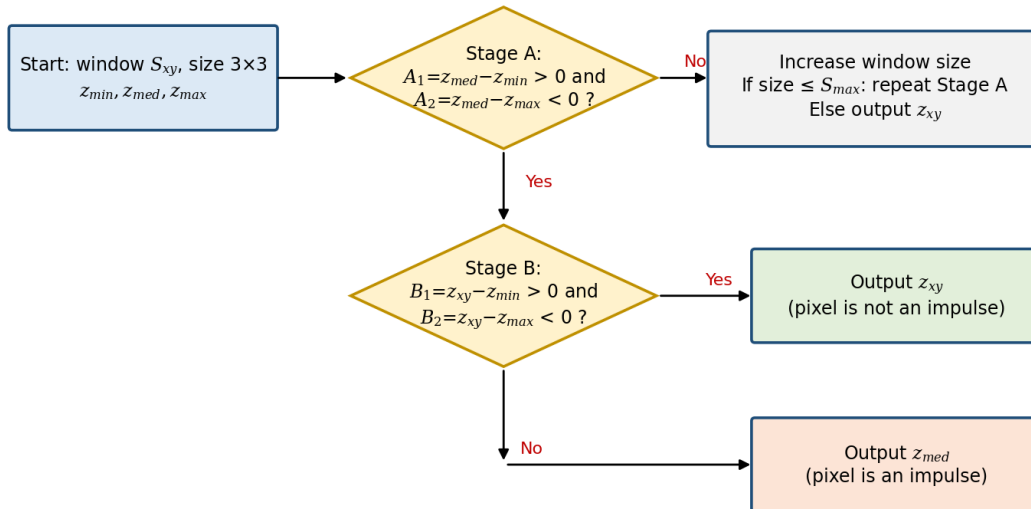


Figure 14: Flowchart of the adaptive median filter.

The adaptive median filter removes salt-and-pepper noise, smooths other non-impulsive noise and reduces the distortion of object boundaries. It tolerates much higher impulse densities than the standard median filter.

5. Restoration of Noisy Images using Frequency Domain Filters

5.1. Principles of Frequency Domain Filtering

By the convolution theorem (Section 2.4.2), filtering an image with a filter of transfer function $H(u, v)$ amounts to multiplying the spectrum of the image by $H(u, v)$. The filtered image is the real part of the inverse transform of the product:

$$\hat{f}(x, y) = \text{Re}\{\mathcal{F}^{-1}[H(u, v) G(u, v)]\} \quad (51)$$

The procedure, shown in Figure 15, is the following:

1. **Zero-padding:** pad the $M \times N$ image to a size $P \times Q$ with $P \geq 2M - 1$ and $Q \geq 2N - 1$ to avoid wraparound error due to the circular convolution of the DFT.
2. **Centring:** multiply the padded image by $(-1)^{x+y}$ so that the zero frequency is at the centre ($P/2, Q/2$) of the spectrum.
3. **Transform:** compute the DFT $G(u, v)$ of the padded image.
4. **Filtering:** multiply $G(u, v)$ by the filter $H(u, v)$, defined in the same centred coordinates and of the same size.
5. **Inverse transform:** compute the inverse DFT, take the real part and multiply again by $(-1)^{x+y}$.
6. **Crop** the top-left $M \times N$ region to recover the filtered image.

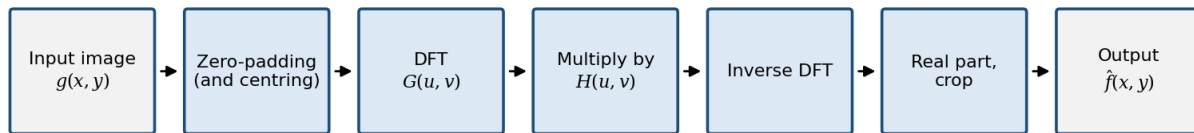


Figure 15: Frequency-domain filtering procedure.

The filters are designed as functions of the distance from the centre of the (centred) spectrum:

$$D(u, v) = \sqrt{\left(u - \frac{P}{2}\right)^2 + \left(v - \frac{Q}{2}\right)^2} \quad (52)$$

A radially symmetric filter has the advantage of treating all orientations alike. Since the DFT of a real image is symmetric, $H(u, v)$ must also be symmetric about the centre so that the output is real. Frequency-domain filters are

particularly convenient when the filter has a large spatial support or when the noise has a known frequency signature.

5.2. Low-Pass, High-Pass, and Band-Pass Filters

A **low-pass filter** attenuates the high frequencies and keeps the low ones: it smooths the image and reduces noise, but blurs the edges. A **high-pass filter** does the opposite: it attenuates the low frequencies and emphasizes the edges and fine details (and the noise). A **band-pass filter** keeps only a band of intermediate frequencies. Three families of low-pass filters with cutoff frequency D_0 are commonly used. The **ideal** low-pass filter cuts all frequencies above D_0 with a sharp transition:

$$H_{\text{ILPF}}(u, v) = \begin{cases} 1 & D(u, v) \leq D_0 \\ 0 & D(u, v) > D_0 \end{cases} \quad (53)$$

The **Butterworth** low-pass filter of order n has a smoother transition, controlled by n , and the **Gaussian** low-pass filter has no sharp transition at all:

$$H_{\text{BLPF}}(u, v) = \frac{1}{1 + [D(u, v)/D_0]^{2n}} \quad (54)$$

$$H_{\text{GLPF}}(u, v) = e^{-D^2(u, v)/(2D_0^2)} \quad (55)$$

The cutoff frequency D_0 of an ideal filter can be related to the fraction of the image power that it retains, where $P_T = \sum_u \sum_v P(u, v)$ is the total power of the spectrum (Section 1.3):

$$\alpha = 100 \left[\frac{\sum_u \sum_{v: D(u, v) \leq D_0} P(u, v)}{P_T} \right] \% \quad (56)$$

The ideal filter has a serious drawback: its spatial impulse response is a sinc function, so that multiplying the spectrum by a sharp cutoff produces **ringing** artifacts (oscillations around edges), a consequence of the Gibbs phenomenon seen in Section 2.1.1. The Butterworth filter reduces ringing as the order n decreases, and the Gaussian filter has no ringing. The **high-pass** filters are obtained from the low-pass ones, and a **band-pass** filter can be built as the difference of two low-pass filters of cutoff $D_2 > D_1$:

$$H_{\text{HP}}(u, v) = 1 - H_{\text{LP}}(u, v) \quad (57)$$

$$H_{\text{BP}}(u, v) = H_{\text{LP}}(u, v; D_2) - H_{\text{LP}}(u, v; D_1) \quad (58)$$

The transfer functions are plotted in Figure 16 and Figure 17 shows their effect on a synthetic image; Table 7 compares the three families.

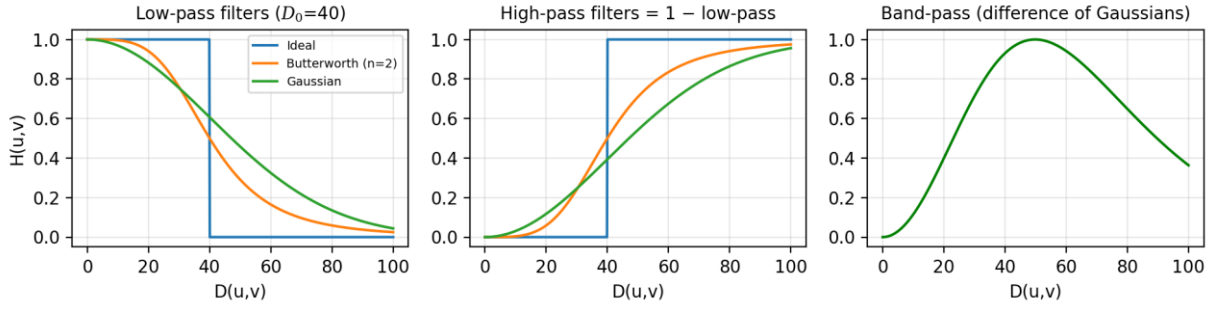


Figure 16: Radial profiles of low-pass, high-pass and band-pass transfer functions.

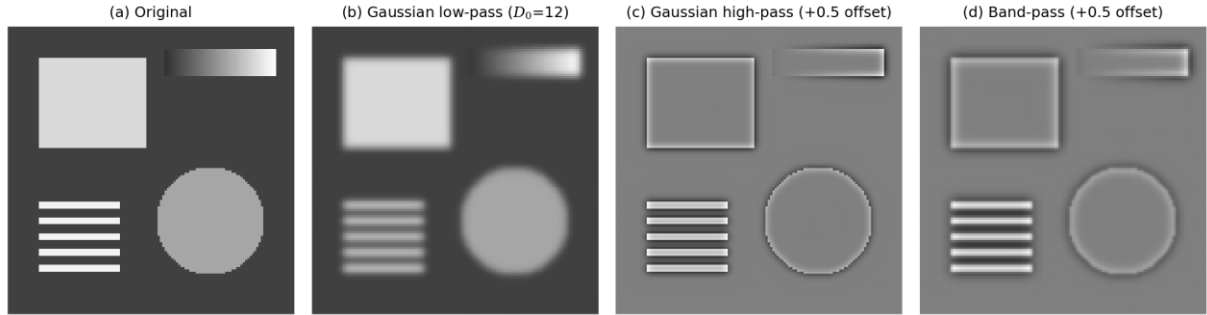


Figure 17: Gaussian low-pass, high-pass and band-pass filtering of a synthetic image (simulation).

Table 7: Comparison of the ideal, Butterworth and Gaussian low-pass filters.

Filter	Transition	Ringings	Parameters
Ideal	Abrupt cut at D_0	Strong	D_0
Butterworth	Smooth, sharper as n increases	Low for small n , increases with n	D_0 , order n
Gaussian	Very smooth	None	D_0 (standard deviation)

5.3. Band-Reject and Notch Filters

5.3.1. Periodic Noise in the Spectrum

Periodic noise is hard to remove in the spatial domain, but its Fourier transform is concentrated in a few spikes. For the sinusoidal interference of Section 3.2.3:

$$N(u, v) = j \frac{AMN}{2} [\delta(u + u_0, v + v_0) - \delta(u - u_0, v - v_0)] \quad (59)$$

The spikes are located visually or automatically in the centred log-spectrum, and a filter is designed to remove them while leaving the rest of the spectrum unchanged.

5.3.2. Band-Reject Filters

A **band-reject filter** removes a ring of frequencies around the origin and is used when the noise frequencies lie on a circle. With a centre D_0 and width W , the ideal, Butterworth and Gaussian versions are:

$$H_{\text{ideal}}(u, v) = \begin{cases} 0 & D_0 - \frac{W}{2} \leq D(u, v) \leq D_0 + \frac{W}{2} \\ 1 & \text{otherwise} \end{cases} \quad (60)$$

$$H_{\text{BW}}(u, v) = \frac{1}{1 + \left[\frac{D(u, v) W}{D^2(u, v) - D_0^2} \right]^{2n}} \quad (61)$$

$$H_{\text{G}}(u, v) = 1 - e^{-\frac{1}{2} \left[\frac{D^2(u, v) - D_0^2}{D(u, v) W} \right]^2} \quad (62)$$

The corresponding **band-pass** filter is $H_{bp} = 1 - H_{br}$; it is mainly used to isolate the noise pattern, whose inverse transform can be analysed or subtracted from the image. The profiles of the three band-reject filters are shown in Figure 18.

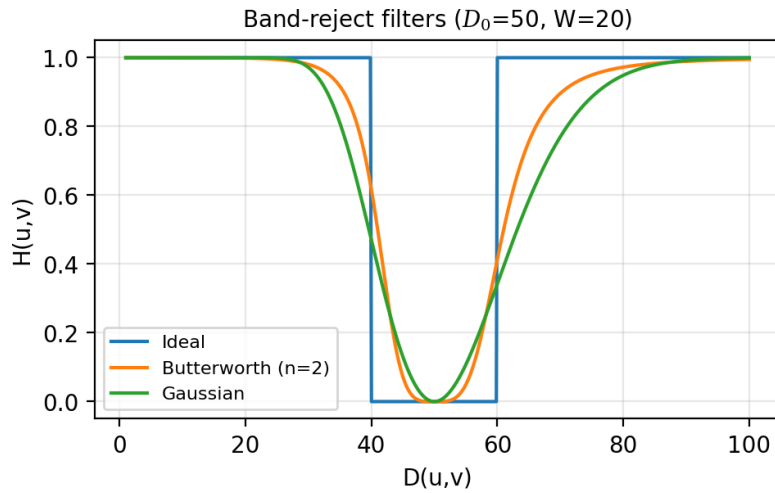


Figure 18: Radial profile of the ideal, Butterworth and Gaussian band-reject filters.

5.3.3. Notch Filters

A **notch reject filter** removes the frequencies in a small neighbourhood of chosen points. Because the spectrum of a real image is symmetric, the notches come in pairs centred at (u_k, v_k) and $(-u_k, -v_k)$. It is built as a product of high-pass filters, one pair per noise spike:

$$H_{\text{NR}}(u, v) = \prod_{k=1}^Q H_k(u, v) H_{-k}(u, v) \quad (63)$$

For Butterworth high-pass notches of radius D_{0k} and order n :

$$H_k(u, v) = \frac{1}{1 + \left[\frac{D_{0k}}{D_k(u, v)} \right]^{2n}}, \quad D_k(u, v)$$

$$= \sqrt{\left(u - \frac{P}{2} - u_k\right)^2 + \left(v - \frac{Q}{2} - v_k\right)^2} \quad (64)$$

and $D_{-k}(u, v)$ is obtained by replacing u_k and v_k by $-u_k$ and $-v_k$. The notch pass filter is $H_{NP}(u, v) = 1 - H_{NR}(u, v)$. To remove periodic noise:

1. Compute the DFT of the (zero-padded) degraded image and display its centred log-spectrum.
2. Locate the pairs of spikes caused by the interference.
3. Build the notch reject filter with one pair of notches per spike.
4. Multiply the spectrum by the filter and compute the inverse DFT (Section 5.1).

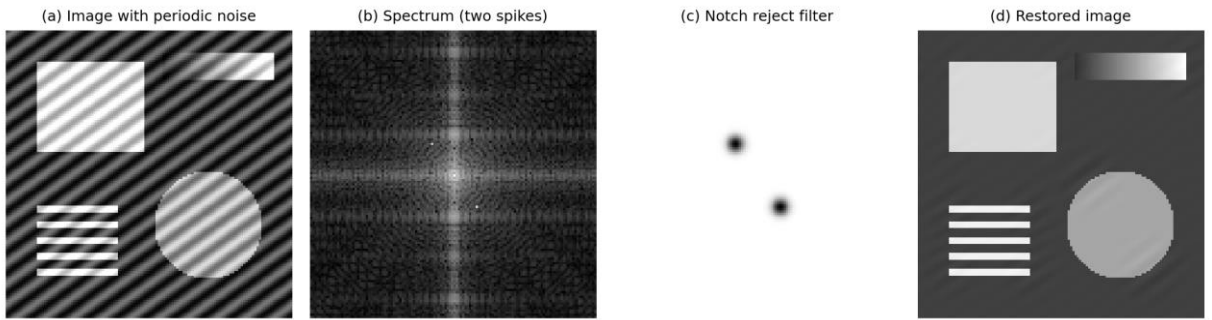


Figure 19: Removal of periodic noise with a notch reject filter on a synthetic image (simulation).

Table 8: Comparison of the band-reject and notch filters.

Filter	Removed frequencies	When to use	Remark
Band-reject	Ring $D_0 \pm W/2$ around the origin	Noise on a circle in the spectrum	Also removes useful image frequencies of the ring
Band-pass	Keeps the ring only	Isolate the noise pattern	$H_{bp} = 1 - H_{br}$
Notch reject	Small neighbourhoods around chosen points (in pairs)	Isolated spikes (stripes, grids)	Least damage to the image content
Notch pass	Keeps the notches only	Extract the noise component	$H_{NP} = 1 - H_{NR}$

6. Image Reconstruction from Projections

In the previous sections the image was observed directly and degraded by noise. In **computed tomography** the object (for example a cross-section of the human body) is never seen directly: a set of one-dimensional **projections** is measured from many directions, and the image of the cross-section must be *reconstructed* from them. Beyond X-ray CT, the same principles are used in PET and SPECT emission tomography, industrial non-destructive testing, electron tomography and geophysical imaging.

6.1. Principles of Computed Tomography (CT)

In X-ray CT, a narrow beam of intensity I_0 crosses the object and reaches a detector with a lower intensity I (Figure 20). According to the Beer–Lambert law, the attenuation depends on the linear attenuation coefficient $\mu(x, y)$ of the tissues along the ray L :

$$I = I_0 e^{-\int_L \mu(x,y) ds} \quad (65)$$

Taking the logarithm gives a measurement p that is a **line integral** of the unknown image $f(x, y) = \mu(x, y)$:

$$p = -\ln \left(\frac{I}{I_0} \right) = \int_L \mu(x, y) ds \quad (66)$$

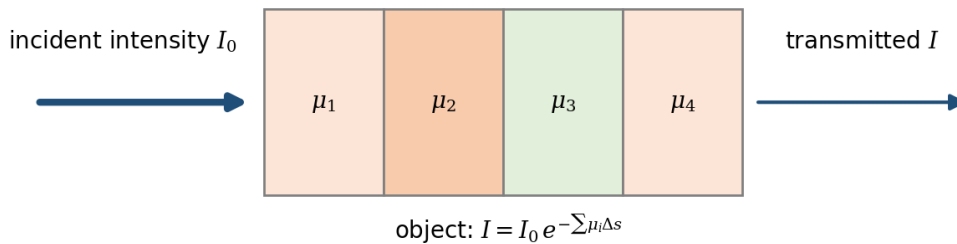


Figure 20: Attenuation of an X-ray beam through successive layers of different attenuation coefficients.

The set of line integrals measured for one viewing angle is a **projection**, and a scan consists in measuring projections for many angles. The reconstructed values are usually expressed in Hounsfield units (HU), which normalize the attenuation with respect to that of water so that water is 0 HU and air is about -1000 HU:

$$\text{HU} = 1000 \frac{\mu - \mu_{\text{water}}}{\mu_{\text{water}}} \quad (67)$$

The geometry of the acquisition determines the reconstruction algorithm (Figure 21 and Table 9). In the **parallel-beam** geometry, the rays of a projection are parallel; a source and a detector array translate and rotate around the object. It is the simplest geometry and most of the reconstruction theory is derived for it. In the **fan-beam** geometry, a point source emits a fan of diverging rays measured by an array of detectors; the source and detectors rotate together around the patient, which gives a much faster acquisition. Fan-beam data can be reconstructed with a specific fan-beam algorithm or by *rebinning* the rays into parallel projections.

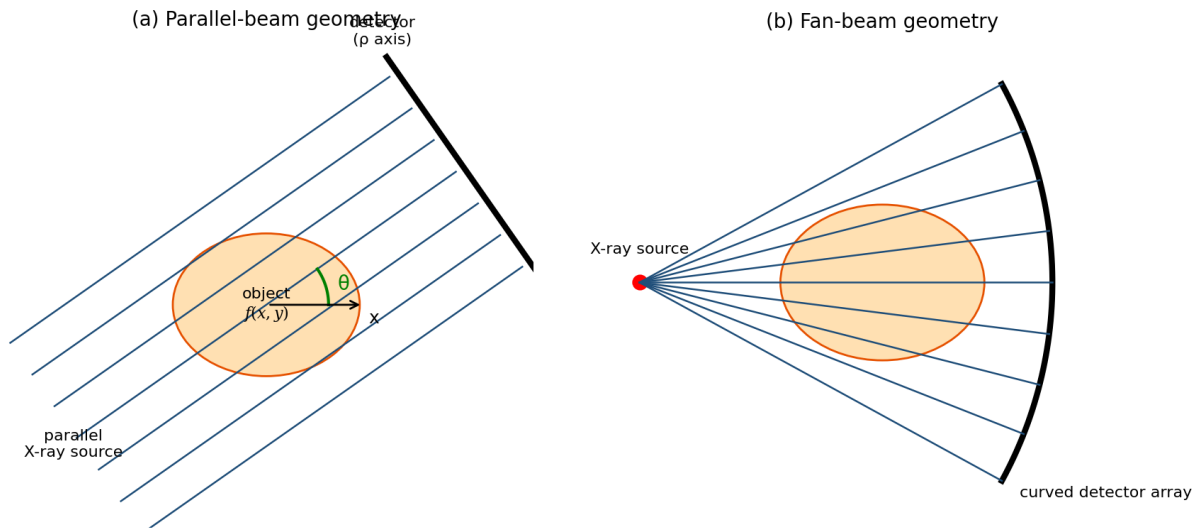


Figure 21: Parallel-beam and fan-beam acquisition geometries.

Table 9: Comparison of the parallel-beam and fan-beam geometries.

Characteristic	Parallel beam	Fan beam
X-ray source	Collimated beam (parallel rays)	Point source emitting a fan of rays
Detectors	Linear array, translated or wide	Flat or curved array facing the source
Movement	Translation and rotation	Rotation only
Acquisition speed	Slow	Fast
Reconstruction	Direct (backprojection, FBP)	Fan-beam FBP, or rebinning to parallel data

6.2. The Radon Transform

A ray in the plane is determined by its angle θ and by its signed distance ρ from the origin. The points (x, y) of the ray satisfy:

$$x \cos \theta + y \sin \theta = \rho \quad (68)$$

The **Radon transform** of the image $f(x, y)$ is the set of all its line integrals. Using the sifting property of the Dirac impulse (Section 1.2.3), it is written as:

$$g(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - \rho) dx dy \quad (69)$$

For a fixed angle θ_k , the function $g(\rho, \theta_k)$ is the projection of the image at that angle. Since $g(\rho, \theta) = g(-\rho, \theta + \pi)$, it is sufficient to consider angles in $[0, \pi)$. The Radon transform is the continuous counterpart of the Hough transform of Chapter 3: the Hough accumulator is the Radon transform of the edge map. The image $g(\rho, \theta)$, displayed with θ on the horizontal axis and ρ on the vertical axis, is the **sinogram** (Figure 22). A single point at (x_0, y_0) is mapped to a sinusoid, which gives the sinogram its name:

$$\begin{aligned} \rho(\theta) &= x_0 \cos \theta + y_0 \sin \theta = \sqrt{x_0^2 + y_0^2} \cos(\theta - \varphi_0), & \varphi_0 \\ &= \arctan \frac{y_0}{x_0} \end{aligned} \quad (70)$$

An arbitrary object is a superposition of such sinusoids: the amplitude of each sinusoid gives the distance of the point from the centre of rotation, and its phase the angular position. The reconstruction problem is the **inversion** of the Radon transform: recover $f(x, y)$ from $g(\rho, \theta)$.

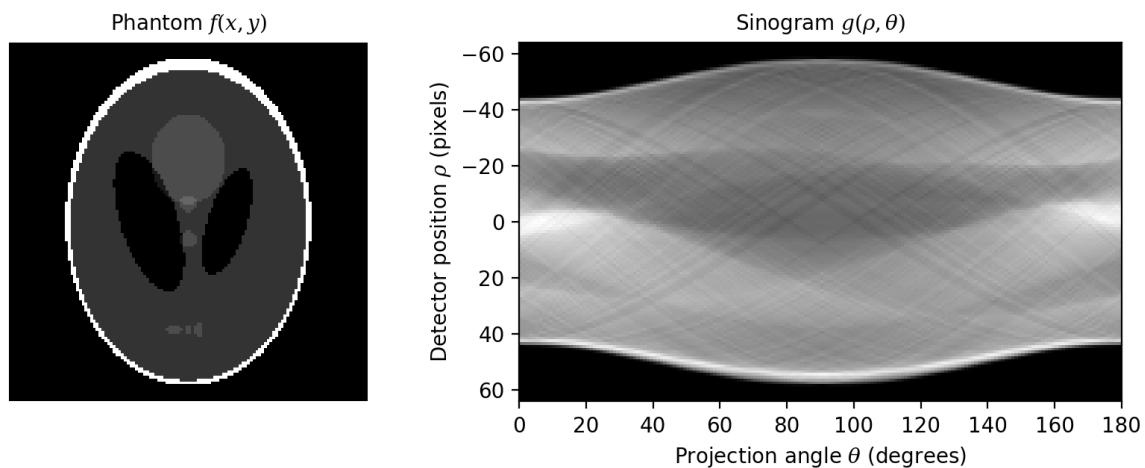


Figure 22: A synthetic phantom and its sinogram, computed with 180 projection angles (simulation).

6.3. Reconstruction Algorithms from Projections (Filtered Backprojection, etc.)

6.3.1. The Fourier Slice Theorem

The Fourier slice theorem links the projections to the 2D Fourier transform of the image and is the foundation of the reconstruction algorithms. Let $G(\omega, \theta)$ be the 1D Fourier transform of the projection $g(\rho, \theta)$ with respect to ρ :

$$G(\omega, \theta) = \int_{-\infty}^{\infty} g(\rho, \theta) e^{-j2\pi\omega\rho} d\rho \quad (71)$$

and $F(u, v)$ the 2D Fourier transform of $f(x, y)$. The theorem states that:

$$G(\omega, \theta) = F(\omega \cos \theta, \omega \sin \theta) \quad (72)$$

In words: **the 1D Fourier transform of the projection at angle θ** is a slice of the 2D Fourier transform of the image, taken along the line through the origin at the same angle θ (Figure 23). To prove it, introduce the rotated coordinates $x = \rho \cos \theta - s \sin \theta$ and $y = \rho \sin \theta + s \cos \theta$. For $(u, v) = (\omega \cos \theta, \omega \sin \theta)$, the exponent $ux + vy$ equals $\omega\rho$, and therefore:

$$\begin{aligned} & F(\omega \cos \theta, \omega \sin \theta) \\ &= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} f(\rho \cos \theta - s \sin \theta, \rho \sin \theta + s \cos \theta) ds \right] e^{-j2\pi\omega\rho} d\rho = G(\omega, \theta) \end{aligned} \quad (73)$$

because the inner integral is exactly the projection $g(\rho, \theta)$. A direct method follows: compute the 1D FFT of each projection, place the results along radial lines in the frequency plane, interpolate onto a Cartesian grid and take the inverse 2D DFT. This *direct Fourier reconstruction* is delicate, because the samples are dense near the origin and sparse at high frequencies, and the polar-to-Cartesian interpolation introduces errors. This observation leads to filtered backprojection.

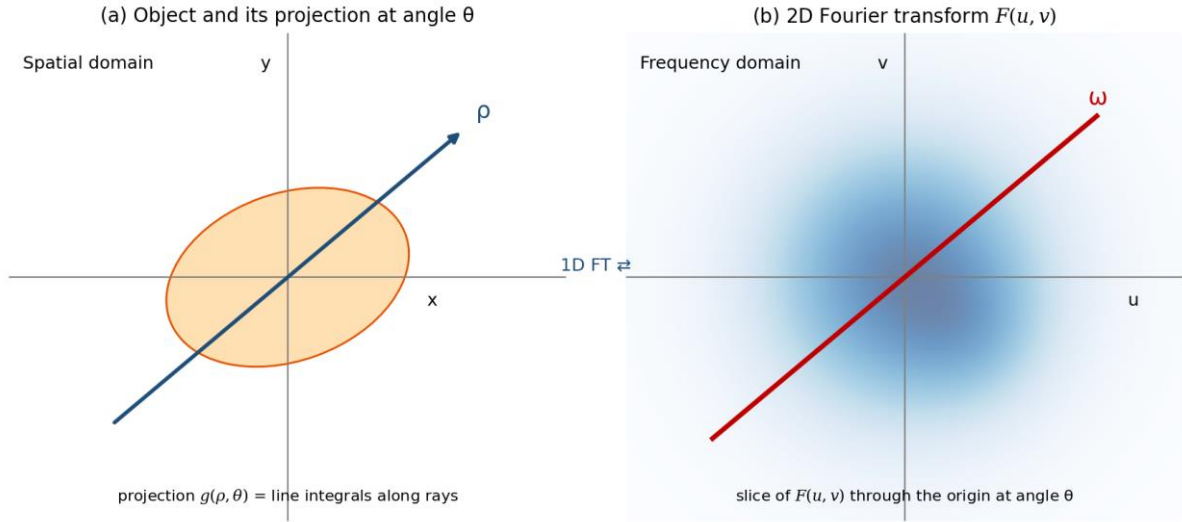


Figure 23: Fourier slice theorem: the 1D Fourier transform of a projection is a radial slice of the 2D Fourier transform of the image.

6.3.2. Simple Backprojection and Its Artifacts

The most intuitive idea is to *smear back* each projection across the image plane along the direction of the rays that produced it, and to sum the contributions of all angles. For a pixel (x, y) , the value of the projection at $\rho = x \cos \theta + y \sin \theta$ is added for each angle:

$$f_b(x, y) = \int_0^\pi g(x \cos \theta + y \sin \theta, \theta) d\theta \quad (74)$$

$$\approx \sum_{i=1}^K g(x \cos \theta_i + y \sin \theta_i, \theta_i) \Delta\theta$$

where K is the number of projections. The result is **not** the original image but a blurred version of it: backprojection is equivalent to convolving the true image with a point spread function proportional to $1/r$:

$$f_b(x, y) = f(x, y) ** \frac{1}{\sqrt{x^2 + y^2}} \quad (75)$$

where $**$ denotes 2D convolution. In the frequency domain, the reconstruction has been multiplied by $1/|\omega|$, a strong low-pass filtering that explains the blur. With a small number of projections, star-shaped streak artifacts also appear around high-contrast objects (Figure 26b and e).

6.3.3. Filtered Backprojection (Ramp and Shepp–Logan Filters)

The blur of simple backprojection can be removed analytically. Writing the inverse 2D Fourier transform in polar coordinates, $u = \omega \cos \theta$ and $v = \omega \sin \theta$,

replacing the 2D transform by the slices of the Fourier slice theorem and using the symmetry $G(-\omega, \theta + \pi) = G(\omega, \theta)$, we obtain:

$$f(x, y) = \int_0^\pi \left[\int_{-\infty}^\infty |\omega| G(\omega, \theta) e^{j2\pi\omega\rho} d\omega \right]_{\rho=x\cos\theta+y\sin\theta} d\theta \quad (76)$$

The factor $|\omega|$ is the Jacobian of the change from Cartesian to polar coordinates. It compensates the over-sampling of the low frequencies near the origin. The algorithm has two steps (Figure 24): **filtering** each projection in the frequency domain with the **ramp filter** $|\omega|$, which gives the filtered projection g_F , then **backprojecting** the filtered projections as in Section 6.3.2:

$$\begin{aligned} g_F(\rho, \theta) &= \mathcal{F}^{-1}\{|\omega| G(\omega, \theta)\}, & f(x, y) \\ &= \int_0^\pi g_F(x\cos\theta + y\sin\theta, \theta) d\theta \end{aligned} \quad (77)$$

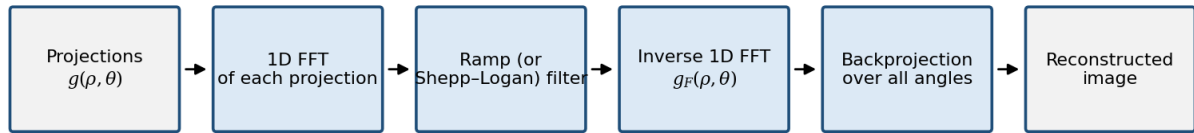


Figure 24: Filtered backprojection algorithm.

The ramp filter amplifies high frequencies and therefore noise. In practice it is truncated at the maximum frequency ω_m (the Nyquist frequency of the detector) and multiplied by a smooth window (Hann, Hamming, cosine, ...) to limit the noise amplification. The **Shepp-Logan filter** is the ramp filter multiplied by a sinc window, which gives a gentle roll-off:

$$H_{SL}(\omega) = \frac{2\omega_m}{\pi} \left| \sin \left(\frac{\pi\omega}{2\omega_m} \right) \right| = |\omega| \operatorname{sinc} \left(\frac{\omega}{2\omega_m} \right), \quad |\omega| \leq \omega_m \quad (78)$$

The filters are compared in Figure 25. Compared with the pure ramp, the Shepp-Logan filter reduces the gain at high frequencies: the reconstruction is less noisy but its resolution is slightly lower. The projections must be zero-padded before the FFT (Section 5.1) to avoid wraparound error. Figure 26 shows the improvement obtained with filtered backprojection on a synthetic phantom.

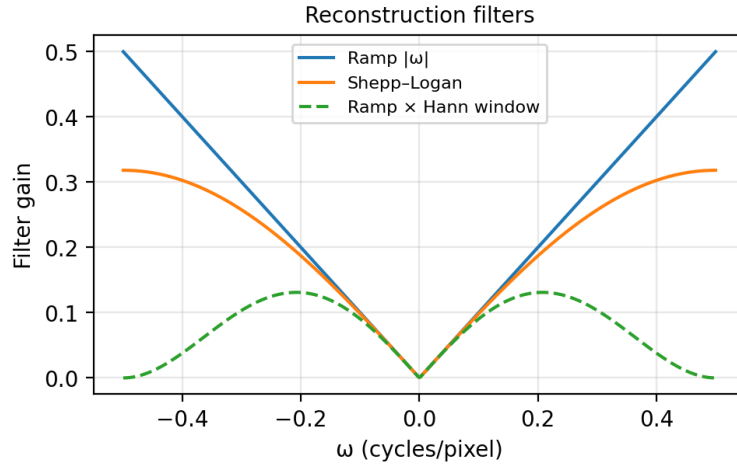


Figure 25: Frequency response of the ramp, Shepp–Logan and windowed ramp filters.

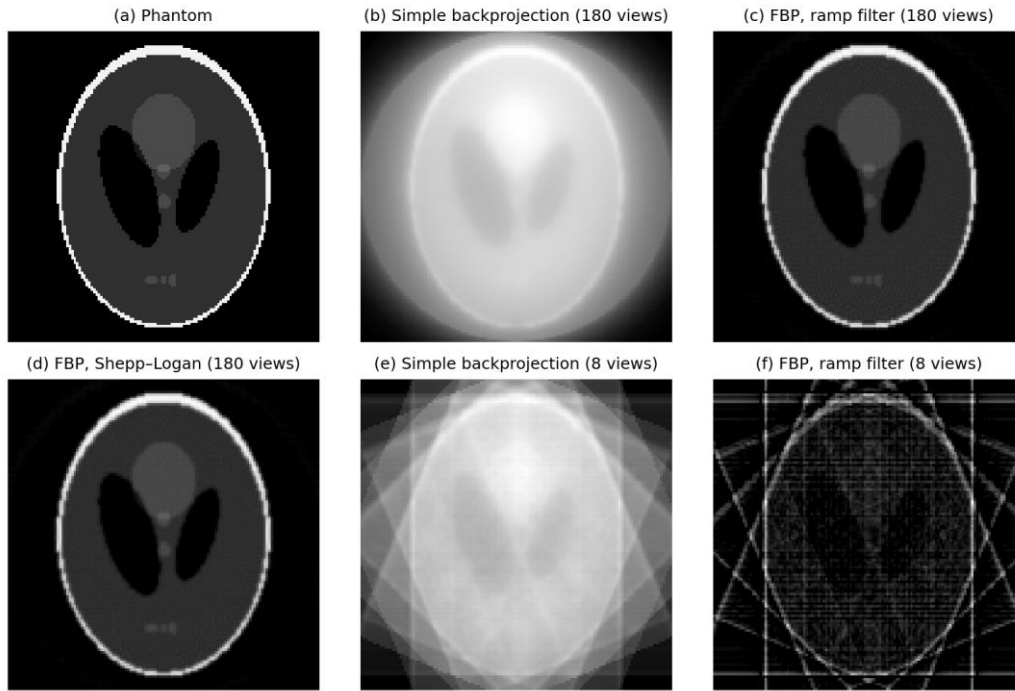


Figure 26: Reconstruction of a synthetic phantom (simulation): simple backprojection versus filtered backprojection, with 180 and with 8 projections. Images are displayed with normalized grey levels.

6.3.4. Introduction to Iterative Reconstruction (ART)

Analytical methods such as FBP require many regularly spaced, low-noise projections. **Iterative methods** instead formulate the reconstruction as the solution of a large system of linear equations. The image is divided into N pixels with unknown values f_j , and each measured ray i gives one equation in which the coefficient a_{ij} represents the contribution (for example the intersection length) of pixel j to the ray i :

$$p_i = \sum_{j=1}^N a_{ij} f_j, \quad i = 1, \dots, I \quad \Leftrightarrow \quad \mathbf{p} = \mathbf{A} \mathbf{f} \quad (79)$$

This system is huge, generally under- or over-determined and noisy, so it cannot be inverted directly. The **Algebraic Reconstruction Technique (ART)**, based on the Kaczmarz method, starts from an initial estimate and processes the rays one after the other. For ray i , it computes the difference between the measured projection and the projection of the current estimate and distributes this error over the pixels crossed by the ray:

$$f_j^{(k+1)} = f_j^{(k)} + \lambda \frac{p_i - \sum_{l=1}^N a_{il} f_l^{(k)}}{\sum_{l=1}^N a_{il}^2} a_{ij} \quad (80)$$

where λ is a relaxation parameter (usually $0 < \lambda < 2$). One pass over all the rays is one iteration; several iterations are performed until the estimate stabilizes, and a non-negativity constraint $f_j \geq 0$ is often enforced. The procedure is:

1. Initialize the image estimate $f^{(0)}$ (for example, all pixels equal to zero).
2. For each ray i , compute the projection of the current estimate, $\sum_l a_{il} f_l^{(k)}$.
3. Compute the error with respect to the measured projection p_i and update the pixels crossed by the ray with Eq. (80).
4. Repeat for all rays (one iteration), apply the constraints and stop when a convergence criterion is satisfied.

Iterative methods are slower than FBP but handle limited-angle or irregularly sampled data, noise and physical constraints much better. Table 10 compares the three algorithms of this section.

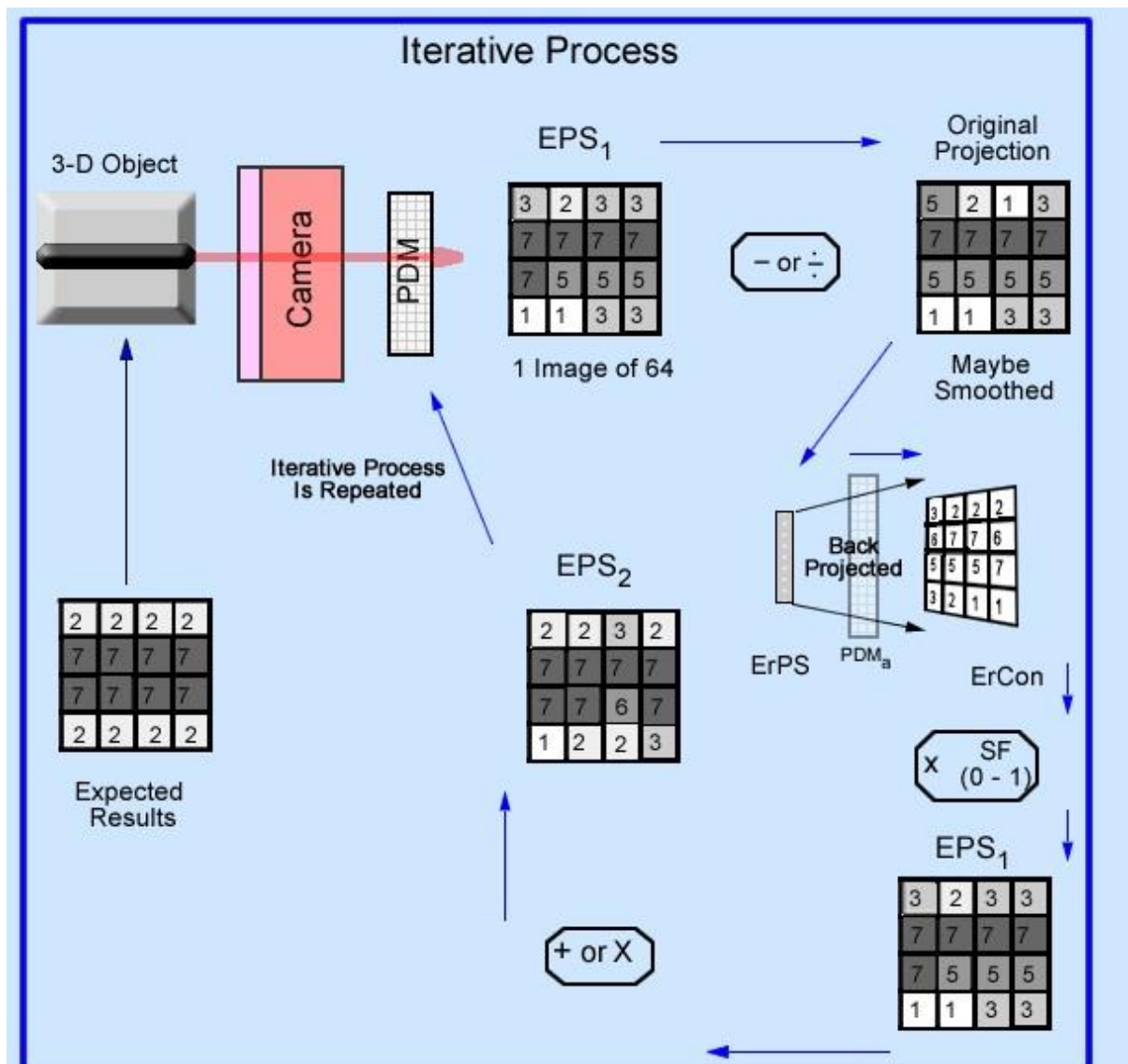


Figure 27 here: Iterative reconstruction (ART): ray-pixel geometry and image estimates after successive iterations

Table 10: Comparison of the tomographic reconstruction algorithms.

Algorithm	Principle	Strengths	Weaknesses
Simple backprojection	Sum of the projections smeared back along the rays	Simple, fast	Blurred result ($1/r$ PSF), streak artifacts
Filtered backprojection (FBP)	Ramp or Shepp-Logan filtering of each projection, then backprojection	Fast, accurate, standard in clinical CT	Needs many regular projections; noise amplification by the ramp
Iterative (ART)	Successive corrections of the image to match each ray equation	Flexible geometry, few views, constraints	Slow; choice of λ and of the stopping rule