Introduction to Image Processing

Lecturer:
Dr. mat. nat. Christian Kehl

Week 5 – Frequency Filters

December 3, 2024

Recap on signal theory principles

Classification of image operations

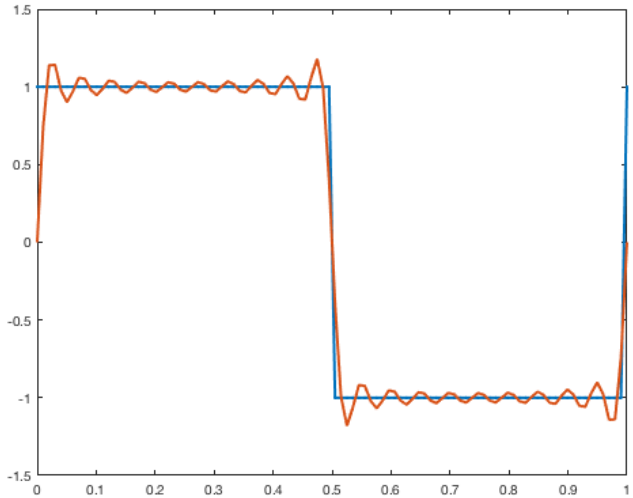
- *Point operation*: output pixel value depends only on corresponding input pixel (e.g. thresholding): $f_{out}(x, y) = \mathcal{O}(\{f_{in}(x, y)\})$ where $\mathcal{O}$ denotes the *grayscale transformation* function.
- *Local operation*: output pixel value depends on neighborhood $\mathcal{B}(x, y)$ of input pixel: $f_{out}(x, y) = \mathcal{O}(\{f_{in}(x', y') : (x', y') \in \mathcal{B}(x, y)\})$.
- *Global operation*: output pixel value depends on all input pixels. Example: (discrete) Fourier transform.
- *Geometric operation*: spatial transformation (scaling, translation, rotation)

Classification of image operations

- *Point operation*: output pixel value depends only on corresponding input pixel (e.g. thresholding): $f_{out}(x, y) = \mathcal{O}(\{f_{in}(x, y)\})$ where $\mathcal{O}$ denotes the *grayscale transformation* function.
- *Local operation*: output pixel value depends on neighborhood $\mathcal{B}(x, y)$ of input pixel: $f_{out}(x, y) = \mathcal{O}(\{f_{in}(x', y') : (x', y') \in \mathcal{B}(x, y)\})$.
- *Global operation*: output pixel value depends on all input pixels. Example: (discrete) Fourier transform.
- *Geometric operation*: spatial transformation (scaling, translation, rotation).

Basics

- signal: superposition of primary frequencies of harmonic function (e.g. *sin* and *cosine*)
- full signal reconstruction: frequency coefficients required → Fourier transform



- Fourier transform requires signal sampling at $\omega = \frac{1}{\Delta T}$
- *perfect* (i.e. non-aliased) reconstruction: sampling width $\Delta T \leqslant \frac{1}{2\mu_{max}}$, with μ_{max} being the maximum frequency in the signal (*Nyquist frequency*)
- smallest detail in signal f relates to highest primary frequency μ_{max} → small features require higher sampling frequency ω

Basics

- digital samples → finite spectrum → **D**FT: from finite set of *samples* $\{f_n\}$ to finite set of *frequency coefficients* $\{F_\mu\}$

$$F_\mu = \sum_{n=0}^{M-1} f_n e^{-i2\pi\mu\frac{n}{M}} \qquad \forall\mu \in \{0..M-1\}$$

- DFT of a convolved signal $f \star h$ equals product of DFTs of individual signals f (input) and h (Dirac sample):

$$\mathcal{F}\{f \star h\} = F_\mu \; H_\mu \qquad \forall\mu \in \{0..M-1\}$$

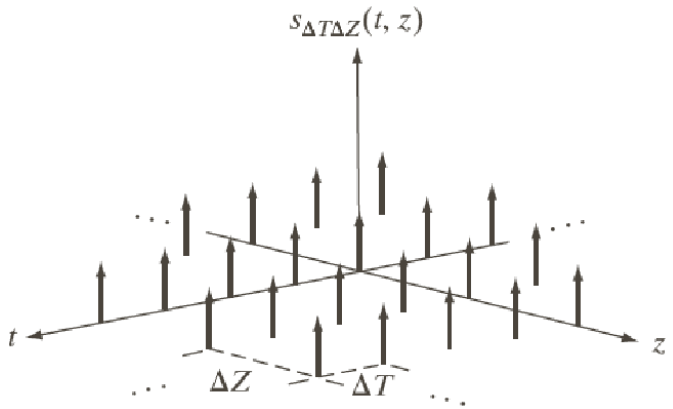
Symbolics & rules

$f(t)$	: signal
$t \in T$	: signal size
$\Delta T \leqslant \frac{1}{2\mu_{max}}$	: sample spacing
μ_{max}	: maximum primary frequency in f
$\Delta\mu = \frac{1}{T}$	: frequency spacing
$\Omega = \frac{1}{\Delta T}$	: size of frequency space
$F_\mu = F(\mu)$	: frequency space of $f(t)$
$s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} \delta(t - n\Delta T)$	: sampling function
$H_\mu = S(\mu)$	: frequency space of $s_{\Delta T}(t)$

2D sampling

A signal $f(x, y)$ can be reconstructed from its frequency space $F(u, v)$ if:

- the amount of sampled frequencies are larger than the highest primary frequencies of the image features:
  $|u| \geqslant \mu_{max}, |v| \geqslant v_{max}$
- the sampling distance adheres to the Nyquist criterion:
  $\Delta T \leqslant \frac{1}{2\mu_{max}}, \Delta Z \leqslant \frac{1}{2v_{max}}$



2D frequency space

Remember that $F(u,v)$ is a *complex* function:

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-i2\pi(u\frac{x}{M}+v\frac{y}{N})} \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

Result: $F(u,v) = R(u,v) + iI(u,v) = |F(u,v)|e^{i\phi(u,v)}|$, with:

- $|F(u,v)| = \sqrt{R^2(u,v) + I^2(u,v)}$ (frequency spectrum)
- $\phi(u,v) = \arctan(\frac{I(u,v)}{R(u,v)})$ (frequency phase angle)

If f is real (i.e. no imaginary part), then the complex conjugate $F^*(u,v)$ is symmetric:

- $F^*(u,v) = F(-u,-v)$
- $|F(u,v)| = |F(-u,-v)|$
- $\phi(u,v) = -\phi(-u,-v)$

2D convolution theorem

For $M \times N$ digital images f and h the 2-D discrete circular convolution is defined by

$$(f \star h)(x,y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n)\, h(x-m, y-n) \;\; \forall x \in \{0..M-1\}, y \in \{0..N-$$

2D convolution theorem

For $M \times N$ digital images f and h the 2-D discrete circular convolution is defined by

$$(f \star h)(x,y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n)\, h(x-m, y-n) \quad \forall x \in \{0..M-1\}, y \in \{0..N-$$

Discrete 2D convolution theorem:

$$\mathcal{F}\{f \star h\}(u,v) = F(u,v)\, H(u,v) \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

2D convolution theorem

For $M \times N$ digital images f and h the 2-D discrete circular convolution is defined by

$$(f \star h)(x,y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n)\, h(x-m, y-n) \quad \forall x \in \{0..M-1\}, y \in \{0..N-$$

Discrete 2D convolution theorem:

$$\mathcal{F}\{f \star h\}(u,v) = F(u,v)\, H(u,v) \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

To compute $f \star h$:

1. compute the Fourier transforms $F(u,v)$ and $H(u,v)$
2. multiply the transforms (pointwise)
3. compute the inverse 2D Fourier transform of the product

Design and application of frequency filters

Frequency domain filtering

1. For $M \times N$ input $f(x, y)$ and $C \times D$ filter $h(x, y)$ apply zero padding to get padded image $f_p(x, y)$ of size $P \times Q$ ($P \geqslant M + C - 1$, $Q \geqslant N + D - 1$).

Frequency domain filtering

1. For $M \times N$ input $f(x, y)$ and $C \times D$ filter $h(x, y)$ apply zero padding to get padded image $f_p(x, y)$ of size $P \times Q$ ($P \geqslant M + C - 1$, $Q \geqslant N + D - 1$).
2. Center its transform by multiplying $f_p(x, y)$ by $(-1)^{x+y}$.

Frequency domain filtering

1. For $M \times N$ input $f(x, y)$ and $C \times D$ filter $h(x, y)$ apply zero padding to get padded image $f_p(x, y)$ of size $P \times Q$ ($P \geqslant M + C - 1$, $Q \geqslant N + D - 1$).
2. Center its transform by multiplying $f_p(x, y)$ by $(-1)^{x+y}$.
3. Compute the DFT of f_p: $F(u, v)$.

Frequency domain filtering

1. For $M \times N$ input $f(x, y)$ and $C \times D$ filter $h(x, y)$ apply zero padding to get padded image $f_p(x, y)$ of size $P \times Q$ ($P \geqslant M + C - 1$, $Q \geqslant N + D - 1$).
2. Center its transform by multiplying $f_p(x, y)$ by $(-1)^{x+y}$.
3. Compute the DFT of f_p: $F(u, v)$.
4. Generate a real symmetric filter $H(u, v)$ of size $P \times Q$ centered at $(P/2, Q/2)$ by zero padding and multiplying $h_p(x, y)$ by $(-1)^{x+y}$.

Frequency domain filtering

1. For $M \times N$ input $f(x, y)$ and $C \times D$ filter $h(x, y)$ apply zero padding to get padded image $f_p(x, y)$ of size $P \times Q$ ($P \geqslant M + C - 1$, $Q \geqslant N + D - 1$).
2. Center its transform by multiplying $f_p(x, y)$ by $(-1)^{x+y}$.
3. Compute the DFT of f_p: $F(u, v)$.
4. Generate a real symmetric filter $H(u, v)$ of size $P \times Q$ centered at $(P/2, Q/2)$ by zero padding and multiplying $h_p(x, y)$ by $(-1)^{x+y}$.
5. Pointwise multiplication: $G(u, v) = H(u, v)F(u, v)$

Frequency domain filtering

1. For $M \times N$ input $f(x, y)$ and $C \times D$ filter $h(x, y)$ apply zero padding to get padded image $f_p(x, y)$ of size $P \times Q$ ($P \geqslant M + C - 1$, $Q \geqslant N + D - 1$).
2. Center its transform by multiplying $f_p(x, y)$ by $(-1)^{x+y}$.
3. Compute the DFT of f_p: $F(u, v)$.
4. Generate a real symmetric filter $H(u, v)$ of size $P \times Q$ centered at $(P/2, Q/2)$ by zero padding and multiplying $h_p(x, y)$ by $(-1)^{x+y}$.
5. Pointwise multiplication: $G(u, v) = H(u, v)F(u, v)$
6. Do the 2-D inverse DFT, take the real part (to remove small imaginary parts due to computational inaccuracy) and undo the centering:

$$g_p(x, y) = \{real[IDFT[G(u, v)]]\}(-1)^{(x+y)}$$

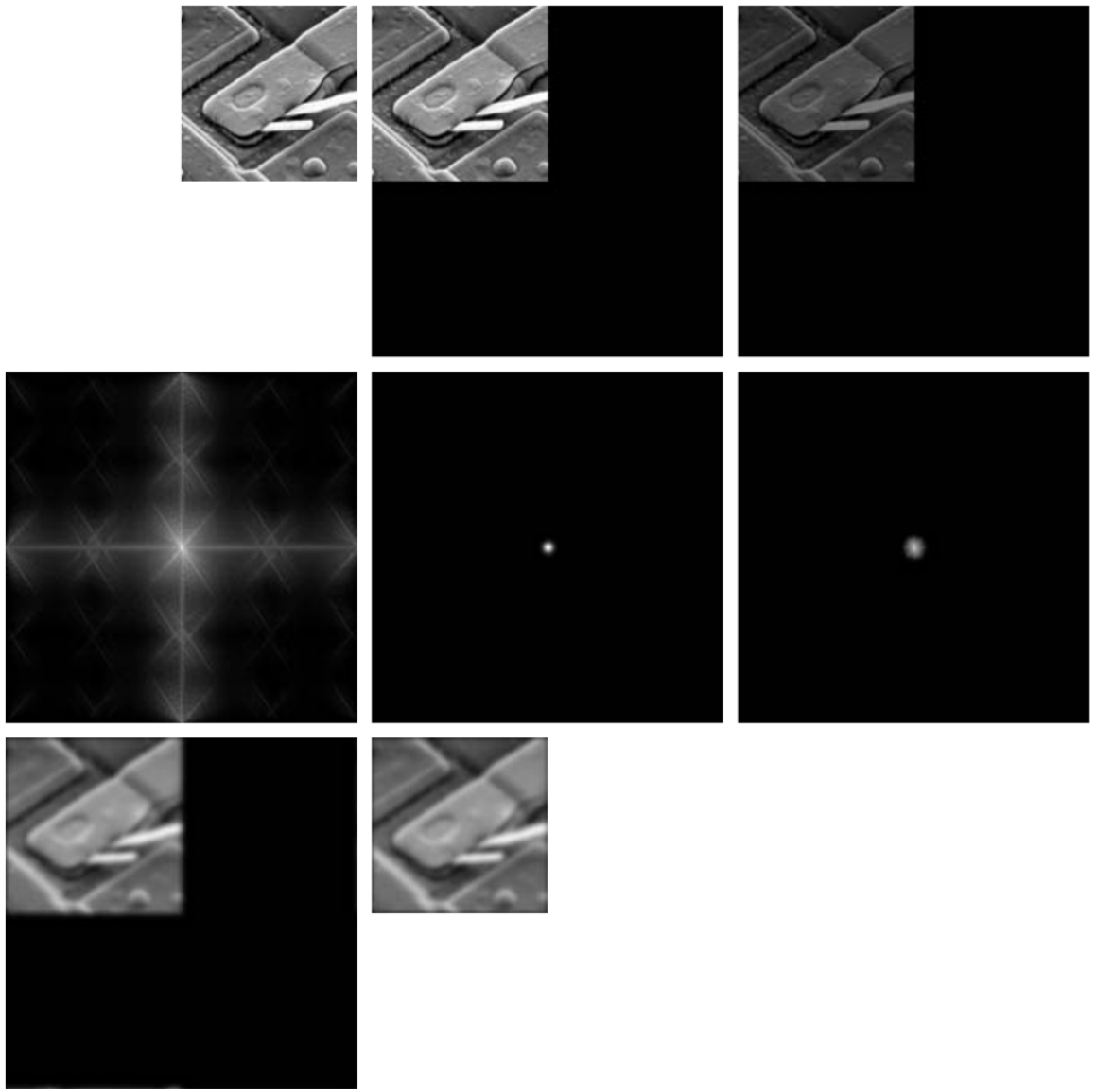
Frequency domain filtering

1. For $M \times N$ input $f(x, y)$ and $C \times D$ filter $h(x, y)$ apply zero padding to get padded image $f_p(x, y)$ of size $P \times Q$ ($P \geqslant M + C - 1$, $Q \geqslant N + D - 1$).
2. Center its transform by multiplying $f_p(x, y)$ by $(-1)^{x+y}$.
3. Compute the DFT of f_p: $F(u, v)$.
4. Generate a real symmetric filter $H(u, v)$ of size $P \times Q$ centered at $(P/2, Q/2)$ by zero padding and multiplying $h_p(x, y)$ by $(-1)^{x+y}$.
5. Pointwise multiplication: $G(u, v) = H(u, v)F(u, v)$
6. Do the 2-D inverse DFT, take the real part (to remove small imaginary parts due to computational inaccuracy) and undo the centering:

$$g_p(x, y) = \{real[IDFT[G(u, v)]]\}(-1)^{(x+y)}$$

7. Obtain $g(x, y)$ by extracting the top left $M \times N$ quadrant from $g_p(x, y)$.

Frequency domain filtering



a b c
d e f
g h

FIGURE 4.36
(a) An $M \times N$ image, f.
(b) Padded image, f_p of size $P \times Q$.
(c) Result of multiplying f_p by $(-1)^{x+y}$.
(d) Spectrum of F_p. (e) Centered Gaussian lowpass filter, H, of size $P \times Q$.
(f) Spectrum of the product HF_p.
(g) g_p, the product of $(-1)^{x+y}$ and the real part of the IDFT of HF_p.
(h) Final result, g, obtained by cropping the first M rows and N columns of g_p.

Filter design

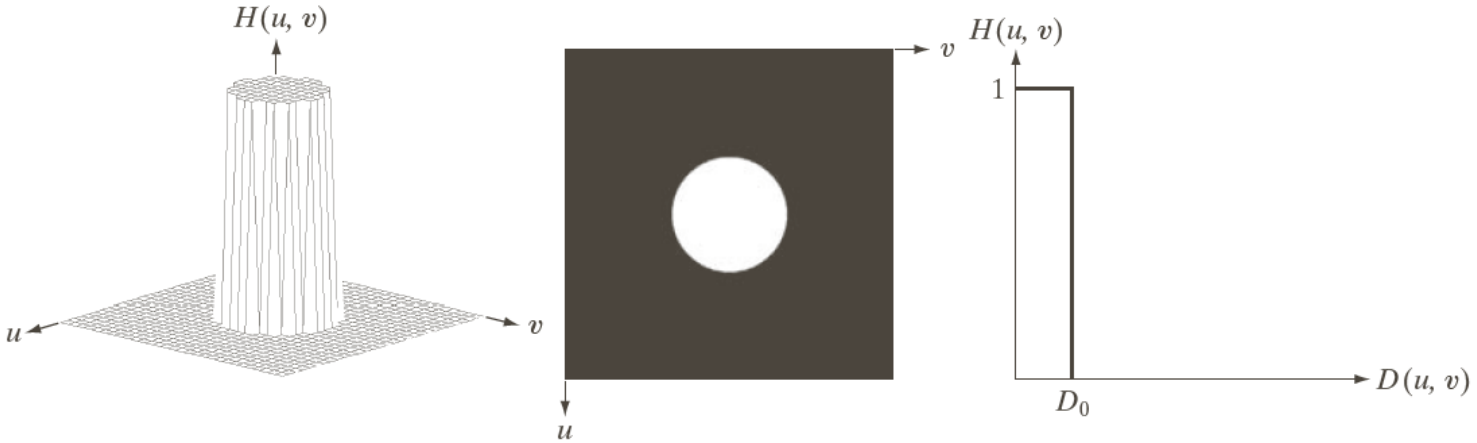
Lowpass filter : suppress high frequencies. Example: box filter ($H(\mu) = 1$ for $0 \leqslant \mu \leqslant \mu_M$, μ_M is the cutoff frequency).

Bandpass filter : pass energy within a given frequency window.

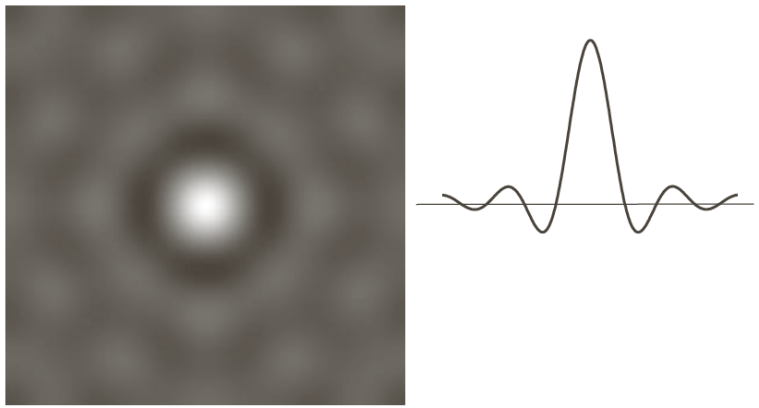
Bandstop filter : suppress energy within a given frequency window.

Highpass filter : suppress low frequencies. (Useful for edge detection).

Ideal lowpass filter (ILPF)

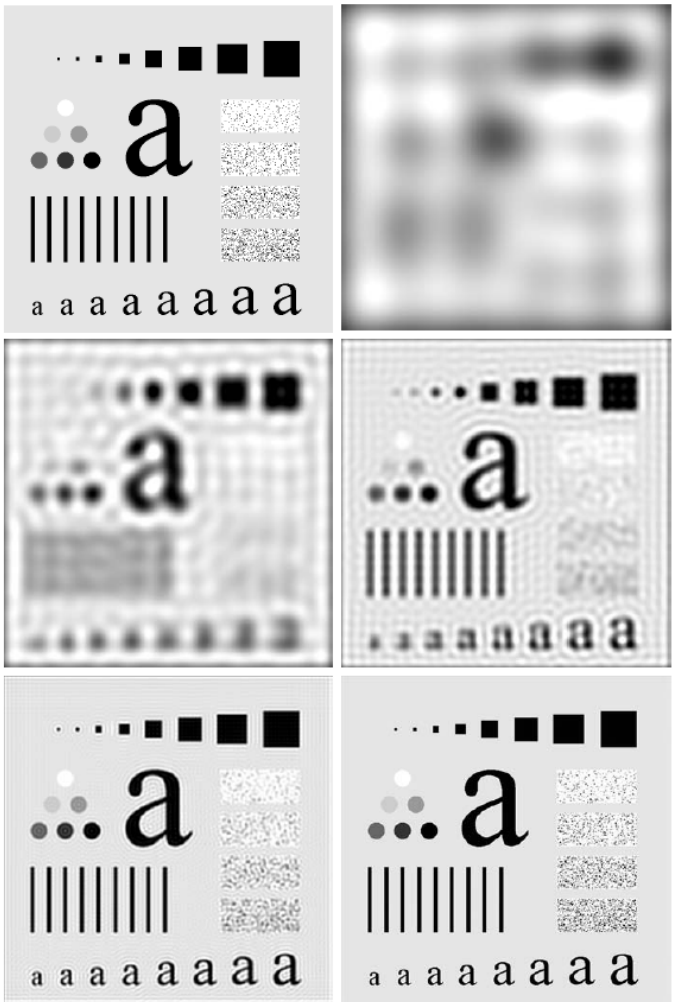


Frequency domain



Spatial domain

Ideal lowpass filter (ILPF)



ILPF with increasing frequency cutoff radius. Note: **ringing artefacs**

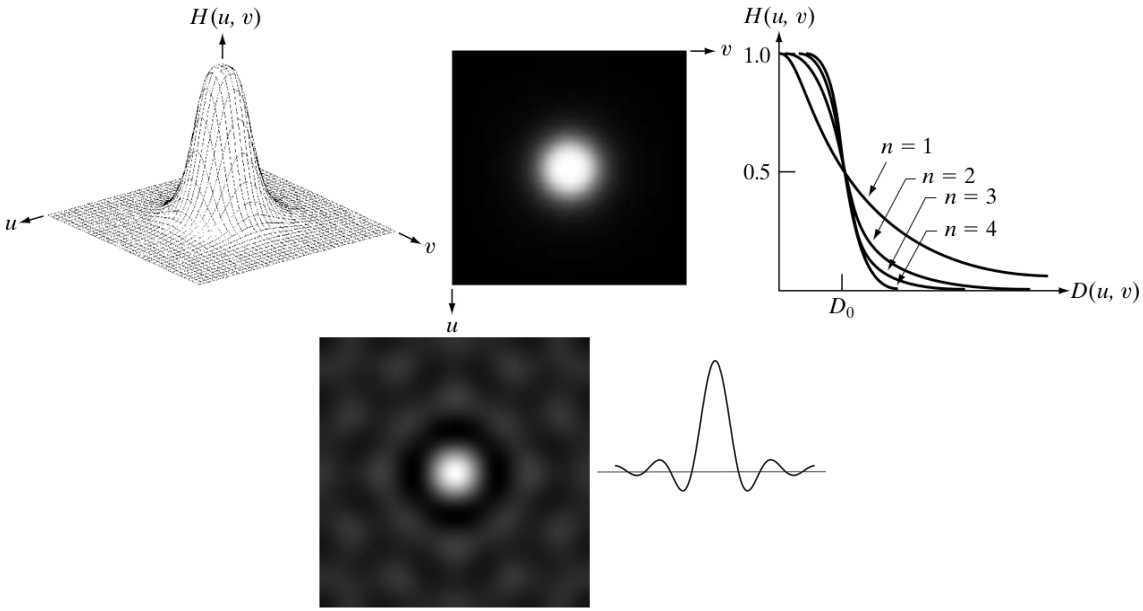
Lowpass frequency filters - Butterworth filter

Ringing artefact → non-smooth spatial transform of filter → **filter design**

$$\text{Low pass}: \quad H(u,v) = \frac{1}{1 + [D(u,v)/D_0]^{2n}}$$

$$\text{Kernel}: \quad D(u,v) = [(u - P/2)^2 + (v - Q/2)^2]^{1/2}, \; D(u,0) = D(u) = u$$

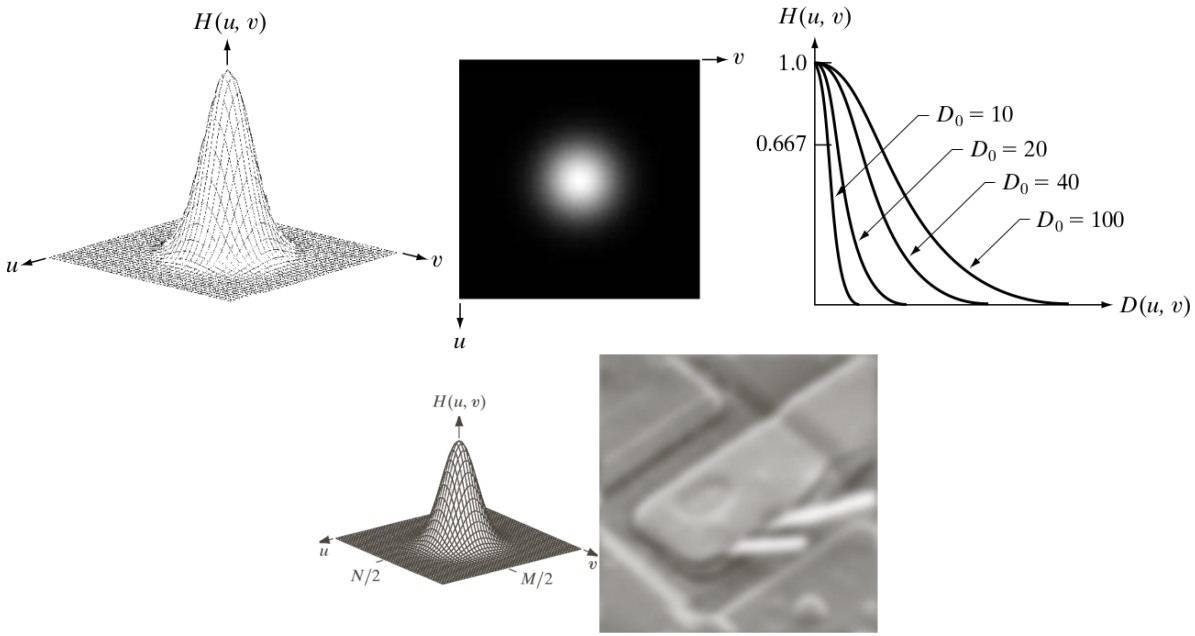
For $\{P, Q\}$, see explanation for image padding before.



Issue: higher-order n reintroduce ringing.

Lowpass frequency filters - Gaussian filter

Low pass : $H(u,v) = e^{-D(u,v)^2/2\sigma^2}$, $h(x,y) = \sqrt{2\pi}\sigma\ e^{-2\pi^2\sigma^2\ D(x,y)^2}$



No ring artefacts, yet also not fully *blocking* high frequencies.

Lowpass frequency filters - Gaussian filter

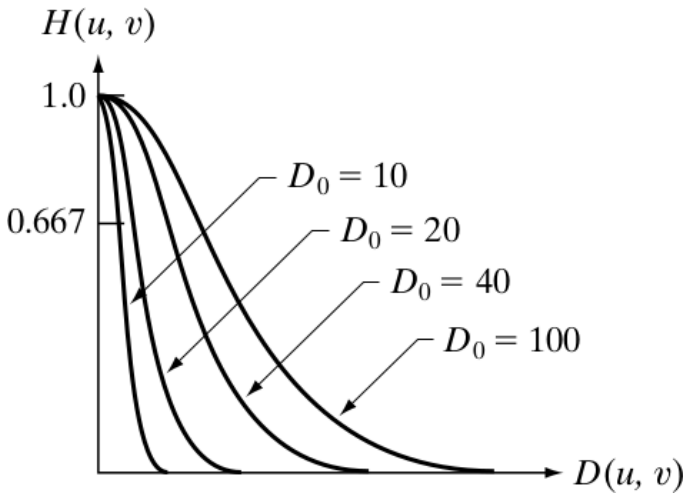
Challenge: determine σ

$$\sigma = D_0$$

$$H(u,v) = e^{-D(u,v)^2/2D_0^2}$$

for $D(u,v) = D_0$:

$$H(u,v) = e^{-D_0^2/2D_0^2} = e^{-\frac{1}{2}} = 0.607$$



For practical purpose:
$\sigma = D_0$ (cutoff frequency)

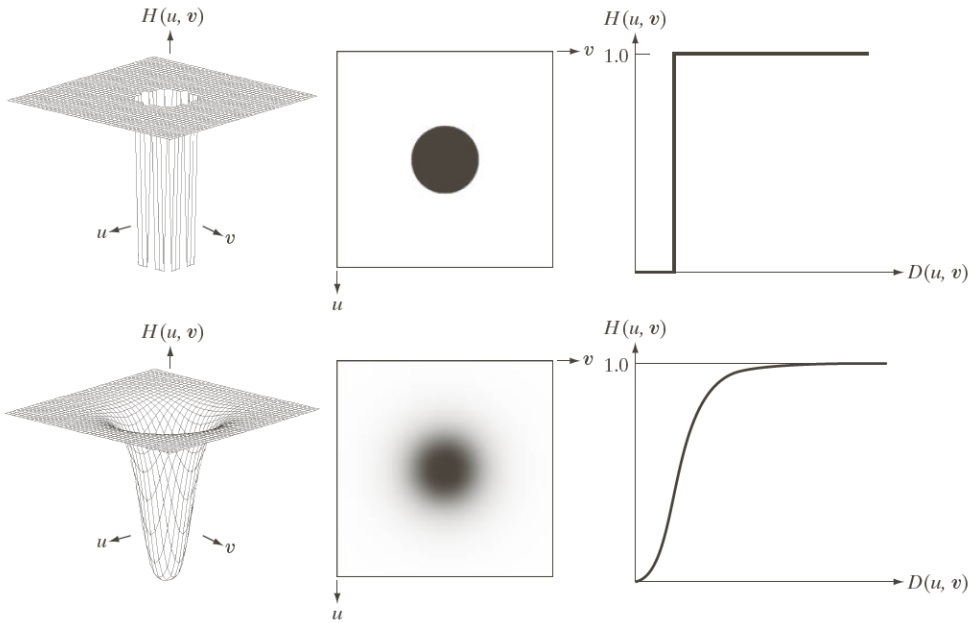
Summary of lowpass filters

TABLE 4.4
Lowpass filters. D_0 is the cutoff frequency and n is the order of the Butterworth filter.

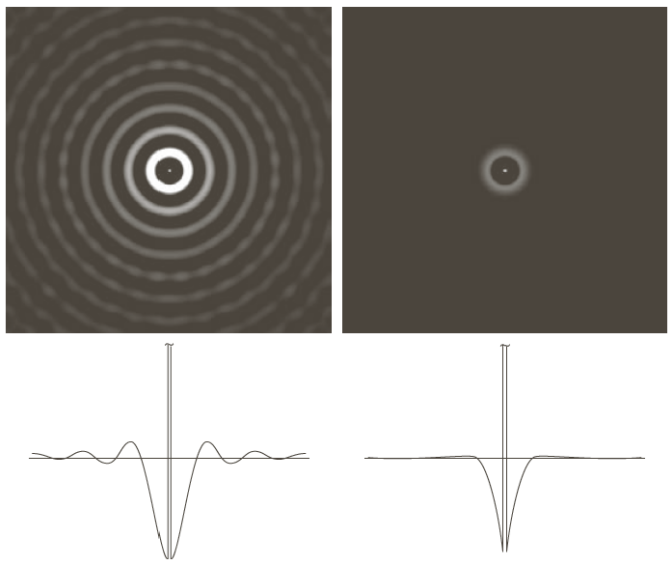
Ideal	Butterworth	Gaussian
$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases}$	$H(u,v) = \dfrac{1}{1 + [D(u,v)/D_0]^{2n}}$	$H(u,v) = e^{-D^2(u,v)/2D_0^2}$

Overview of common lowpass filters.

Highpass frequency filters



Frequency domain.
Top: ideal (IHPF); bottom: Butterworth.



Spatial domain.
Left: ideal (IHPF);
right: Butterworth.

Ideal highpass filter (IHPF)

IHPF with increasing frequency radius.

$$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \leqslant D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases}$$

Highpass frequency filters - Butterworth

Butterworth filter with increasing frequency radius.

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

Highpass frequency filters - Laplacian

Laplacian edge filter in the frequency domain.

$$H(u,v) = -4\pi^2(u^2+v^2) = -4\pi^2\ D^2(u,v)$$

Highpass frequency filters - Gaussians

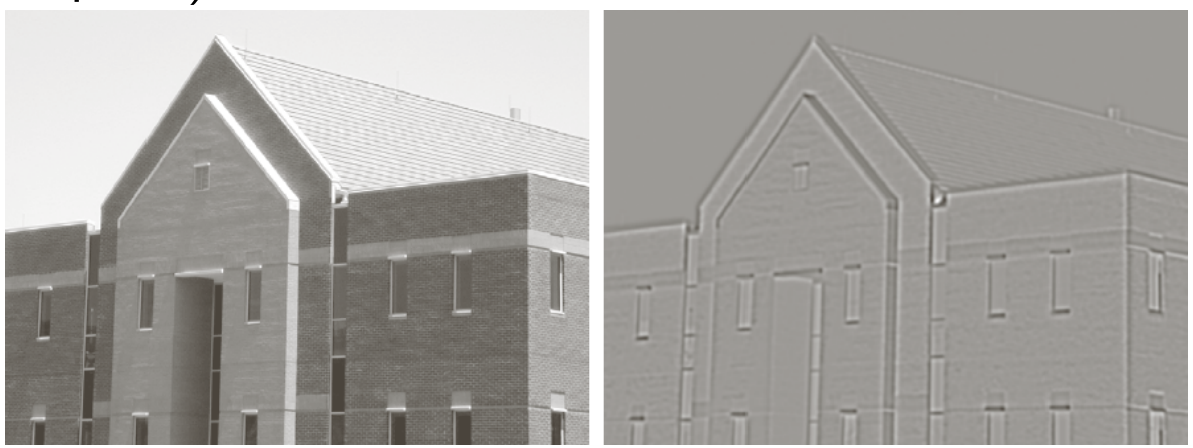
Common Gaussian: $H(u, v) = 1 - e^{-D^2(u,v)/2D_0^2}$

Highpass frequency filters - Gaussians

Common Gaussian: $H(u, v) = 1 - e^{-D^2(u,v)/2D_0^2}$
Laplace-of-Gaussian (i.e. *Marr-Hildreth* filter):

$$K_\sigma(x, y) = -\frac{1}{\pi\sigma^4}\left(1 - \frac{r^2}{2\sigma^2}\right)e^{-\frac{r^2}{2\sigma^2}}$$
$$r^2 = x^2 + y^2$$

(spatial description)



Summary of highpass filters

TABLE 4.5
Highpass filters. D_0 is the cutoff frequency and n is the order of the Butterworth filter.

Ideal	Butterworth	Gaussian
$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases}$	$H(u,v) = \dfrac{1}{1 + [D_0/D(u,v)]^{2n}}$	$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2}$

Overview of common highpass filters.

Bandreject (or Bandstop) filter

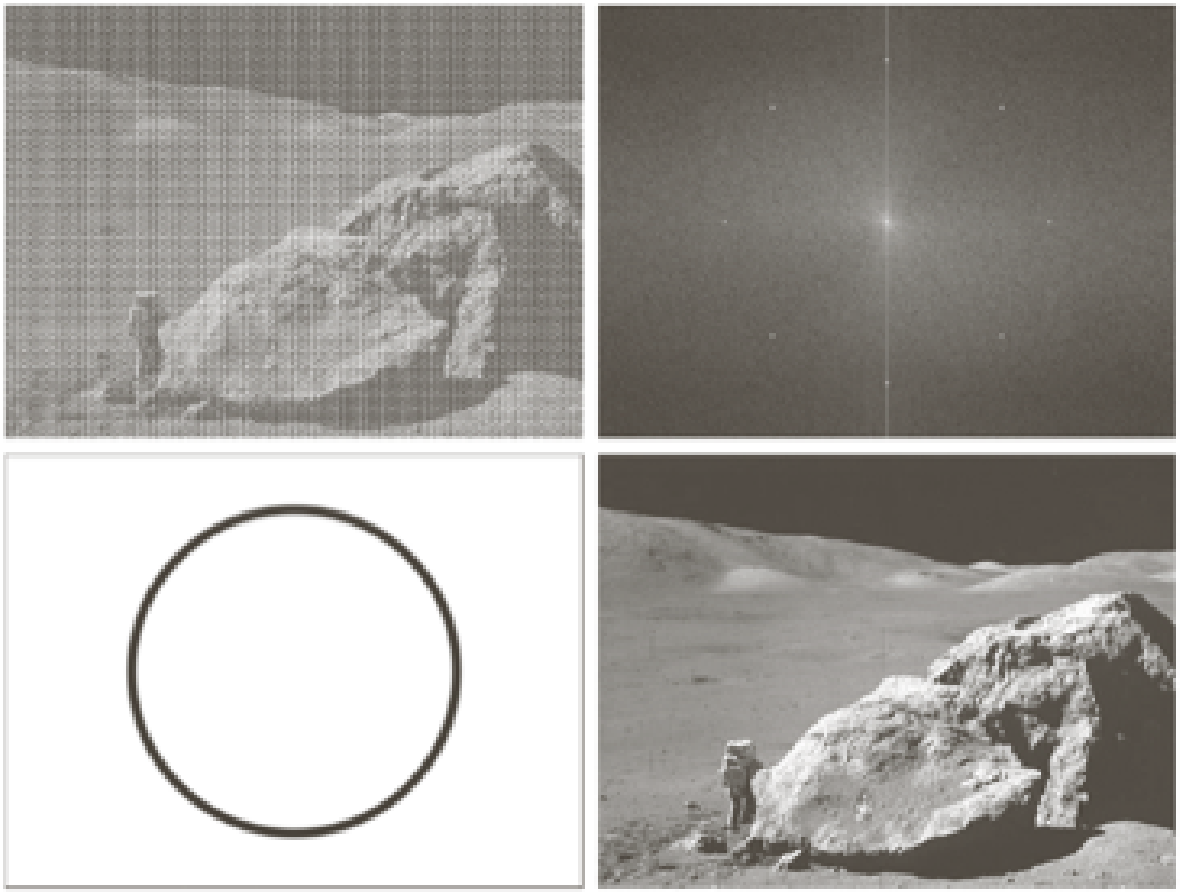
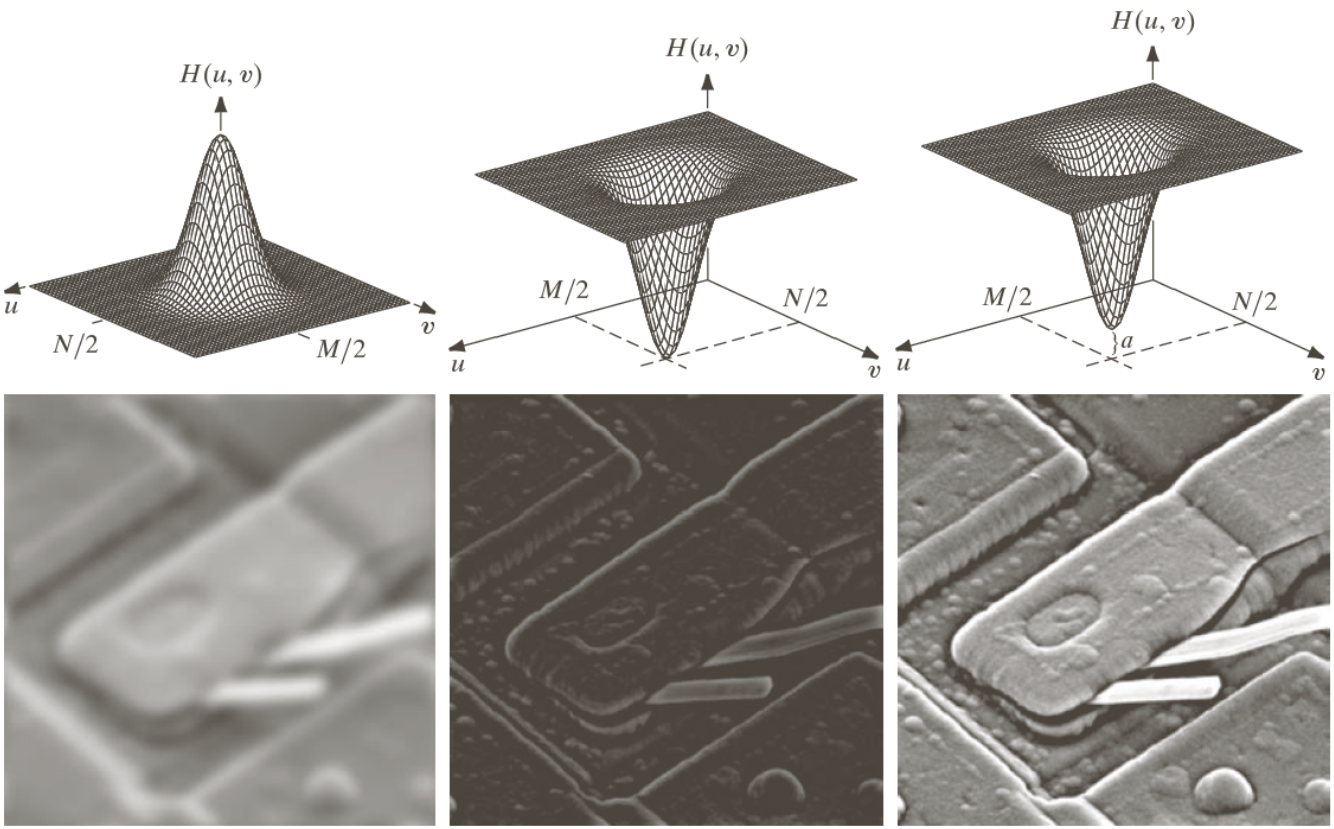


Figure: (a) Image corrupted by sinusoidal noise. (b) Spectrum. (c) Butterworth bandreject filter. (d) Filtered image.

Bandpass filter - Difference-of-Gaussian

$$H(u,v) = Ae^{-D(u,v)^2/2\sigma_1^2} - Be^{-Du,v)^2/2\sigma_2^2} \quad \text{for } a = A - B, A > B$$



Other frequency transformations

Observation 1: We can represent an image as accumulated frequency coefficients.

Other frequency transformations

Observation 1: We can represent an image as accumulated frequency coefficients.

Observation 2: The frequency space is defined by its basic *wave function*.

Other frequency transformations

Observation 1: We can represent an image as accumulated frequency coefficients.
Observation 2: The frequency space is defined by its basic *wave function*.
Assumption: There is more than *one* wave function to span a frequency space over $\{M, N\}$.

Other frequency transformations

Observation 1: We can represent an image as accumulated frequency coefficients.
Observation 2: The frequency space is defined by its basic *wave function*.
Assumption: There is more than *one* wave function to span a frequency space over $\{M, N\}$.

- Fourier function: $F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$
- basic wave function $r(x,y,u,v) = f(x,y) \rightarrow F(u,v)$: $e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$

Other frequency transformations

Observation 1: We can represent an image as accumulated frequency coefficients.
Observation 2: The frequency space is defined by its basic *wave function*.
Assumption: There is more than *one* wave function to span a frequency space over $\{M, N\}$.

- Fourier function: $F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$
- basic wave function $r(x,y,u,v) = f(x,y) \rightarrow F(u,v)$: $e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$

Transformations & basic wave functions:

- Fourier

Other frequency transformations

Observation 1: We can represent an image as accumulated frequency coefficients.
Observation 2: The frequency space is defined by its basic *wave function*.
Assumption: There is more than *one* wave function to span a frequency space over $\{M, N\}$.

- Fourier function: $F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$
- basic wave function $r(x,y,u,v) = f(x,y) \rightarrow F(u,v)$: $e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$

Transformations & basic wave functions:

- Fourier
- Walsh-Hadamard

Other frequency transformations

Observation 1: We can represent an image as accumulated frequency coefficients.
Observation 2: The frequency space is defined by its basic *wave function*.
Assumption: There is more than *one* wave function to span a frequency space over $\{M, N\}$.

- Fourier function: $F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$
- basic wave function $r(x,y,u,v) = f(x,y) \rightarrow F(u,v)$: $e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$

Transformations & basic wave functions:

- Fourier
- Walsh-Hadamard
- Sine & **Cosine**

Other frequency transformations

Observation 1: We can represent an image as accumulated frequency coefficients.
Observation 2: The frequency space is defined by its basic *wave function*.
Assumption: There is more than *one* wave function to span a frequency space over $\{M, N\}$.

- Fourier function: $F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$
- basic wave function $r(x,y,u,v) = f(x,y) \rightarrow F(u,v)$: $e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$

Transformations & basic wave functions:

- Fourier
- Walsh-Hadamard
- Sine & **Cosine**
- Wavelets $\rightarrow$ Gabor, Daubechies, Haar, Shannon, Spline, etc.

Other frequency transformations

Observation 1: We can represent an image as accumulated frequency coefficients.
Observation 2: The frequency space is defined by its basic *wave function*.
Assumption: There is more than *one* wave function to span a frequency space over $\{M, N\}$.

- Fourier function: $F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$
- basic wave function $r(x,y,u,v) = f(x,y) \rightarrow F(u,v)$: $e^{-i2\pi(u\frac{x}{M} + v\frac{y}{N})}$

Transformations & basic wave functions:

- Fourier
- Walsh-Hadamard
- Sine & **Cosine**
- Wavelets $\rightarrow$ Gabor, Daubechies, Haar, Shannon, Spline, etc.

Question: How can we use those base functions ?

Frequency transformations - Base functions

Fourier example:

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-i2\pi(u\frac{x}{M}+v\frac{v}{N})} \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

Reformulation of the *forward, discrete transform*

$$T(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-i2\pi(u\frac{x}{M}+v\frac{v}{N})} \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

$$T(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} g(x,y)r(x,y,u,v) \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

Frequency transformations - Base functions

Fourier example:

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-i2\pi(u\frac{x}{M}+v\frac{v}{N})} \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

Reformulation of the *forward, discrete transform*

$$T(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-i2\pi(u\frac{x}{M}+v\frac{v}{N})} \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

$$T(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} g(x,y)r(x,y,u,v) \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

The *inverse, discrete transform* respectively:

$$g(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} T(u,v)s(x,y,u,v) \quad \forall x \in \{0..M-1\}, y \in \{0..N-1\}$$

Frequency transformations - Base functions

Fourier example:

$$
\begin{aligned}
r(x,y,u,v) &= e^{-j2\pi(u\frac{x}{M}+v\frac{y}{N})} \\
&= e^{-j2\pi u\frac{x}{M}} e^{-j2\pi v\frac{y}{N}} \\
&= r_1(x,u)r_2(y,v) \\
s(x,y,u,v) &= \frac{1}{M}e^{j2\pi x\frac{u}{M}}\frac{1}{N}e^{j2\pi y\frac{v}{N}} \qquad = \frac{1}{NM}e^{j2\pi(x\frac{u}{M}+y\frac{v}{N})}
\end{aligned}
$$

Frequency transformations - Base functions

Fourier example:

$$
\begin{aligned}
r(x,y,u,v) &= e^{-j2\pi(u\frac{x}{M}+v\frac{y}{N})} \\
&= e^{-j2\pi u\frac{x}{M}} e^{-j2\pi v\frac{y}{N}} \\
&= r_1(x,u)r_2(y,v) \\
s(x,y,u,v) &= \frac{1}{M}e^{j2\pi x\frac{u}{M}}\frac{1}{N}e^{j2\pi y\frac{v}{N}} \qquad = \frac{1}{NM}e^{j2\pi(x\frac{u}{M}+y\frac{v}{N})}
\end{aligned}
$$

Next step: let us replace $r(x,y,u,v)$ and $s(x,y,u,v)$ with another base function.

Frequency transforms - Cosines

$$T(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} g(x,y) r(x,y,u,v) \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

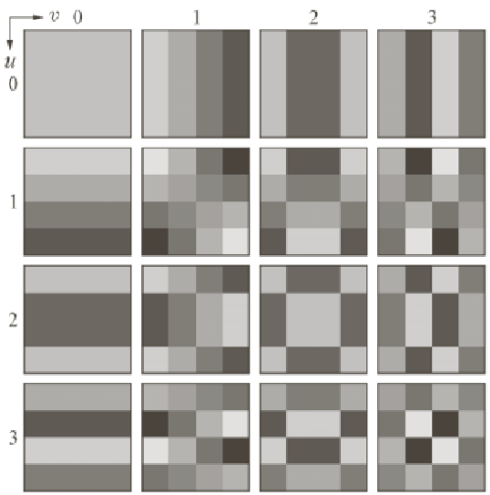
$$g(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} T(u,v) s(x,y,u,v) \quad \forall x \in \{0..M-1\}, y \in \{0..N-1\}$$

Frequency transforms - Cosines

$$T(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} g(x,y)r(x,y,u,v) \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

$$g(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} T(u,v)s(x,y,u,v) \quad \forall x \in \{0..M-1\}, y \in \{0..N-1\}$$

Cosine transform: $r(x,y,u,v) = s(x,y,u,v)$

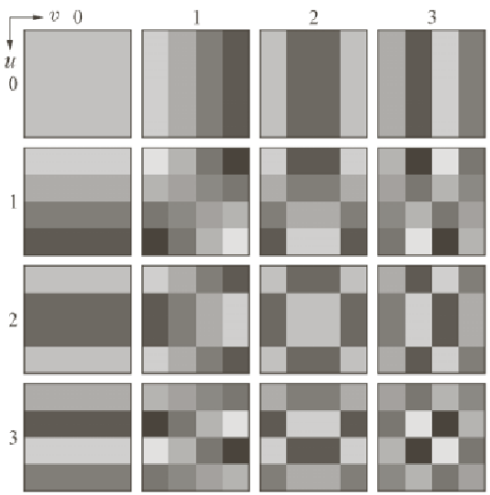


- time-domain 1D: $cos(2\pi\omega(t))$
- actual r_1: $cos(\frac{(2x+1)u\pi}{2M})$

Frequency transforms - Cosines

$$T(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} g(x,y)r(x,y,u,v) \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

$$g(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} T(u,v)s(x,y,u,v) \quad \forall x \in \{0..M-1\}, y \in \{0..N-1\}$$



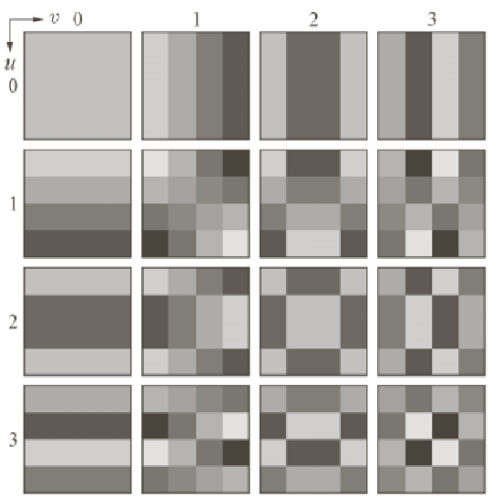
Cosine transform: $r(x,y,u,v) = s(x,y,u,v)$

- time-domain 1D: $cos(2\pi\omega(t))$
- actual r_1: $cos(\frac{(2x+1)u\pi}{2M})$
- complete $r(x,y,u,v)$, $s(x,y,u,v) = \alpha(u)\alpha(v)cos\left[\frac{(2x+1)u\pi}{2M}\right]cos\left[\frac{(2y+1)v\pi}{2N}\right]$

Frequency transforms - Cosines

$$T(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} g(x,y)r(x,y,u,v) \quad \forall u \in \{0..M-1\}, v \in \{0..N-1\}$$

$$g(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} T(u,v)s(x,y,u,v) \quad \forall x \in \{0..M-1\}, y \in \{0..N-1\}$$



Cosine transform: $r(x,y,u,v) = s(x,y,u,v)$

- time-domain 1D: $cos(2\pi\omega(t))$
- actual r_1: $cos(\frac{(2x+1)u\pi}{2M})$
- complete $r(x,y,u,v)$, $s(x,y,u,v) = \alpha(u)\alpha(v)cos\left[\frac{(2x+1)u\pi}{2M}\right]cos\left[\frac{(2y+1)v\pi}{2N}\right]$

$$\alpha(u) = \begin{cases} \sqrt{1/M} & \text{for } u = 0 \\ \sqrt{2/M} & \text{for } u > 0 \end{cases}$$

($\alpha(v)$ analogous)

Discrete Cosine Transform - Applications

Where can we use those transformations ?

1. high- and lowpass filtering $\rightarrow T(u, v)$ are still all *frequency* transforms with coefficients.

Discrete Cosine Transform - Applications

Where can we use those transformations ?

1. high- and lowpass filtering $\rightarrow T(u, v)$ are still all *frequency* transforms with coefficients.
2. simplify or reduce computation

Discrete Cosine Transform - Applications

Where can we use those transformations ?

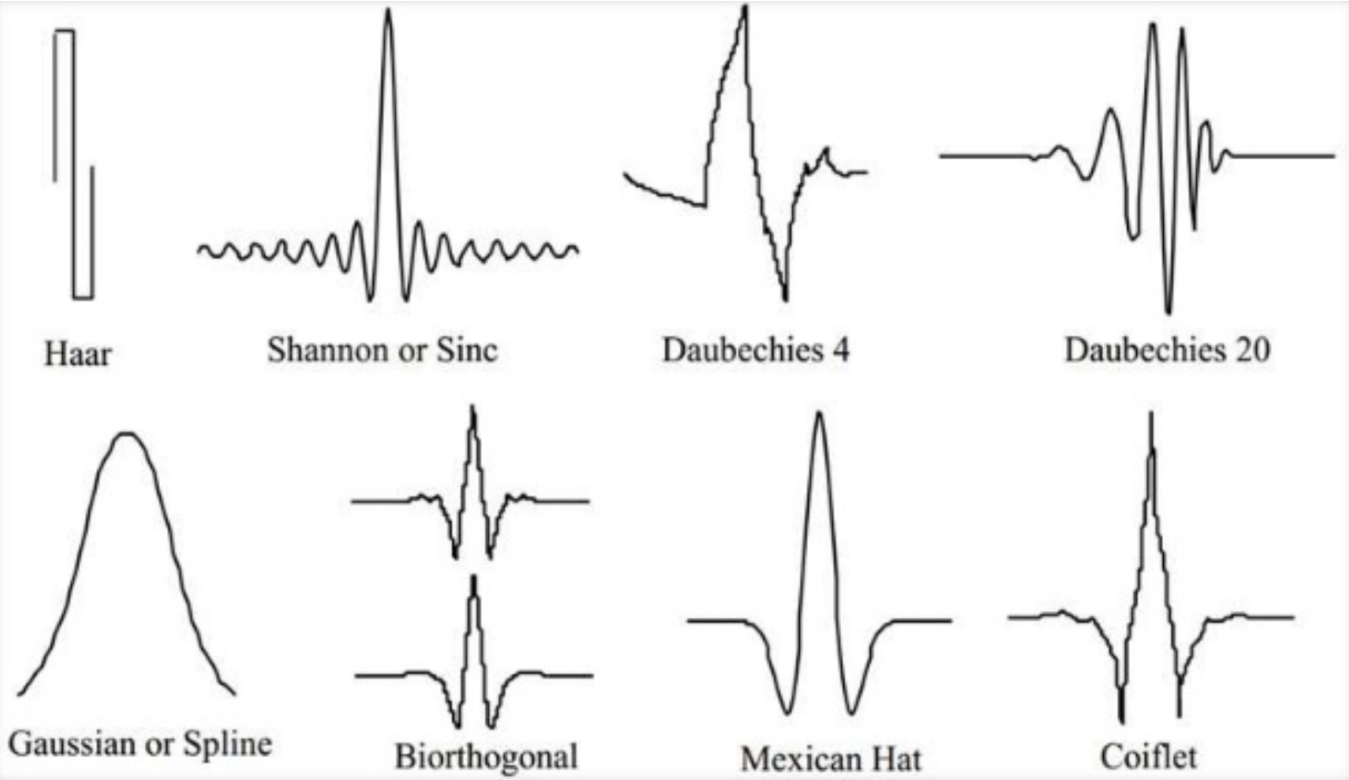
1. high- and lowpass filtering $\rightarrow$ $T(u, v)$ are still all *frequency* transforms with coefficients.
2. simplify or reduce computation
3. image *compression* $\rightarrow$ see this next time.

Discrete Cosine Transform - Applications

Where can we use those transformations ?

1. high- and lowpass filtering $\rightarrow$ $T(u, v)$ are still all *frequency* transforms with coefficients.
2. simplify or reduce computation
3. image *compression* $\rightarrow$ see this next time.

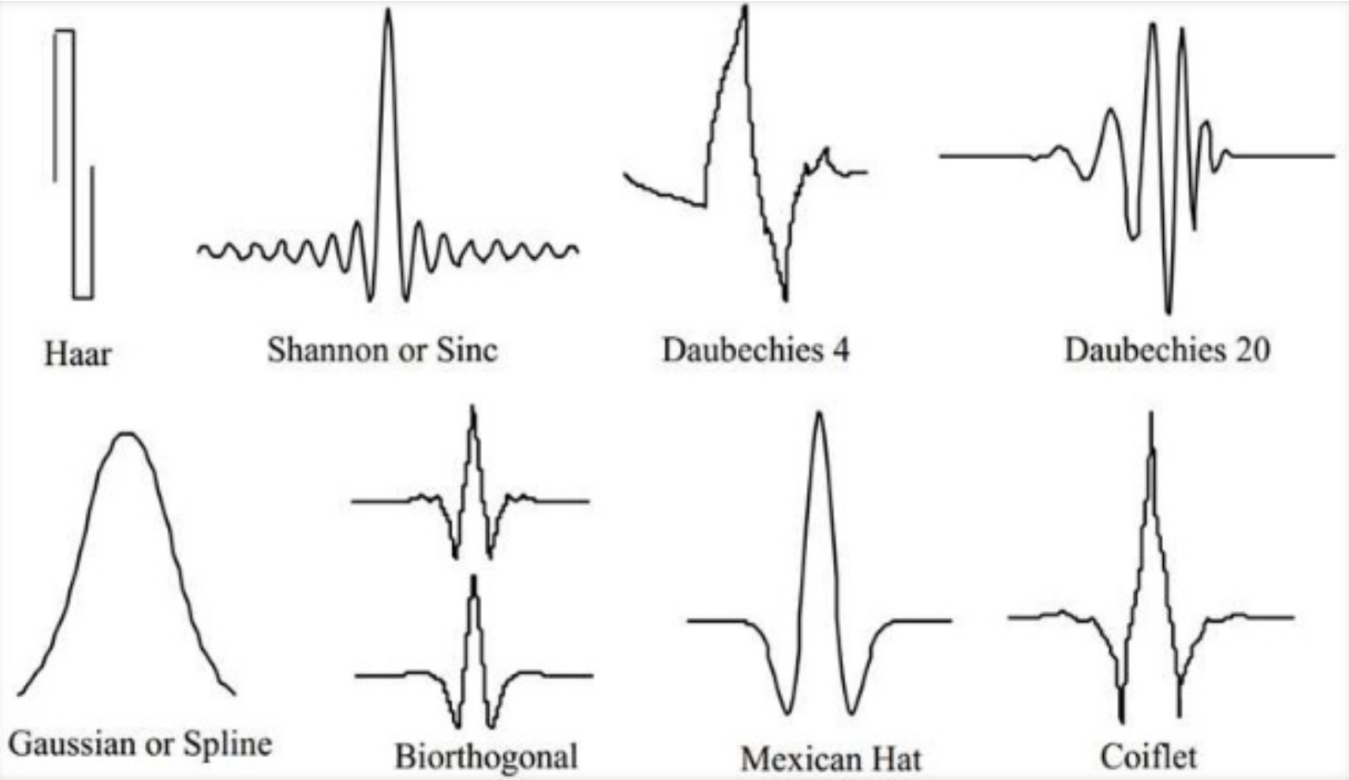
Frequency transformations - Wavelet bases



application depends on:

- the use of the coefficients (filter, store, reconstruct)

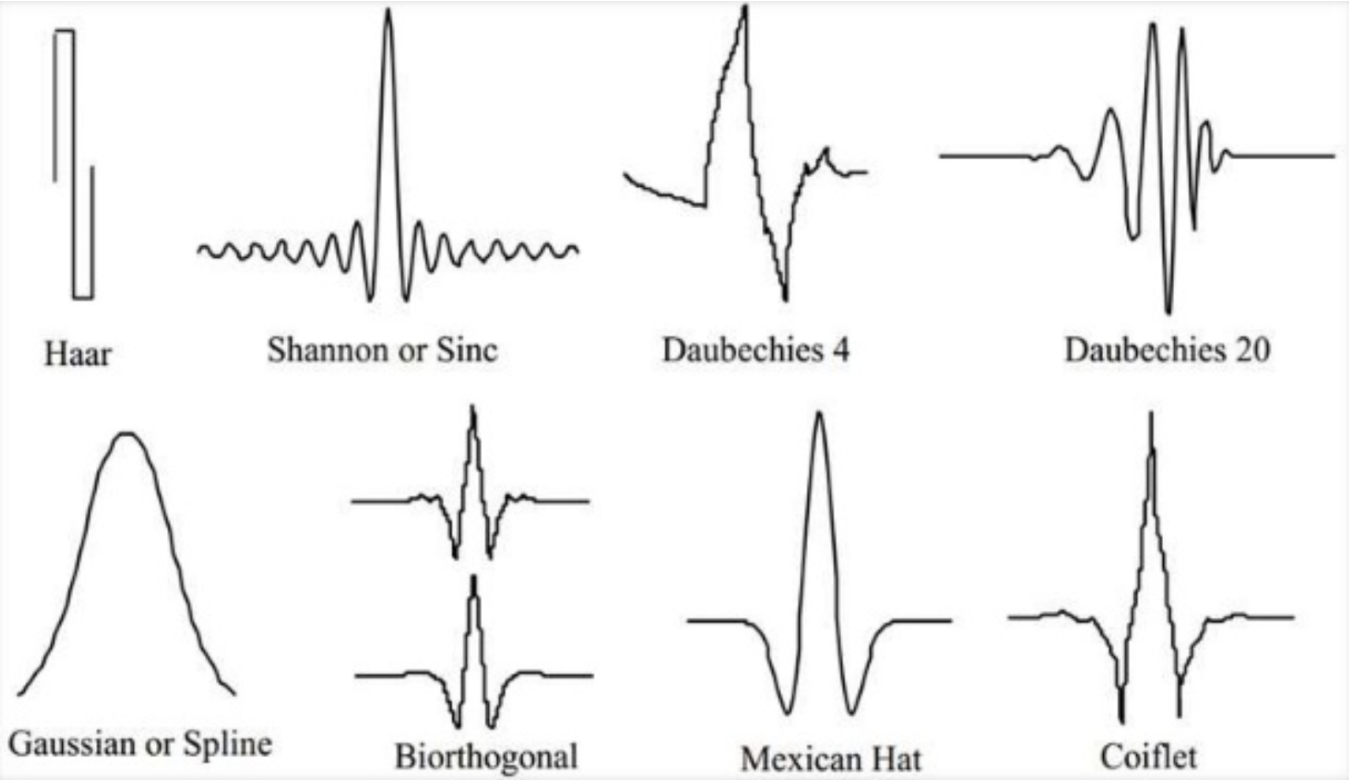
Frequency transformations - Wavelet bases



application depends on:

- the use of the coefficients (filter, store, reconstruct)
- required precision: double, float, half-float, integer, byte

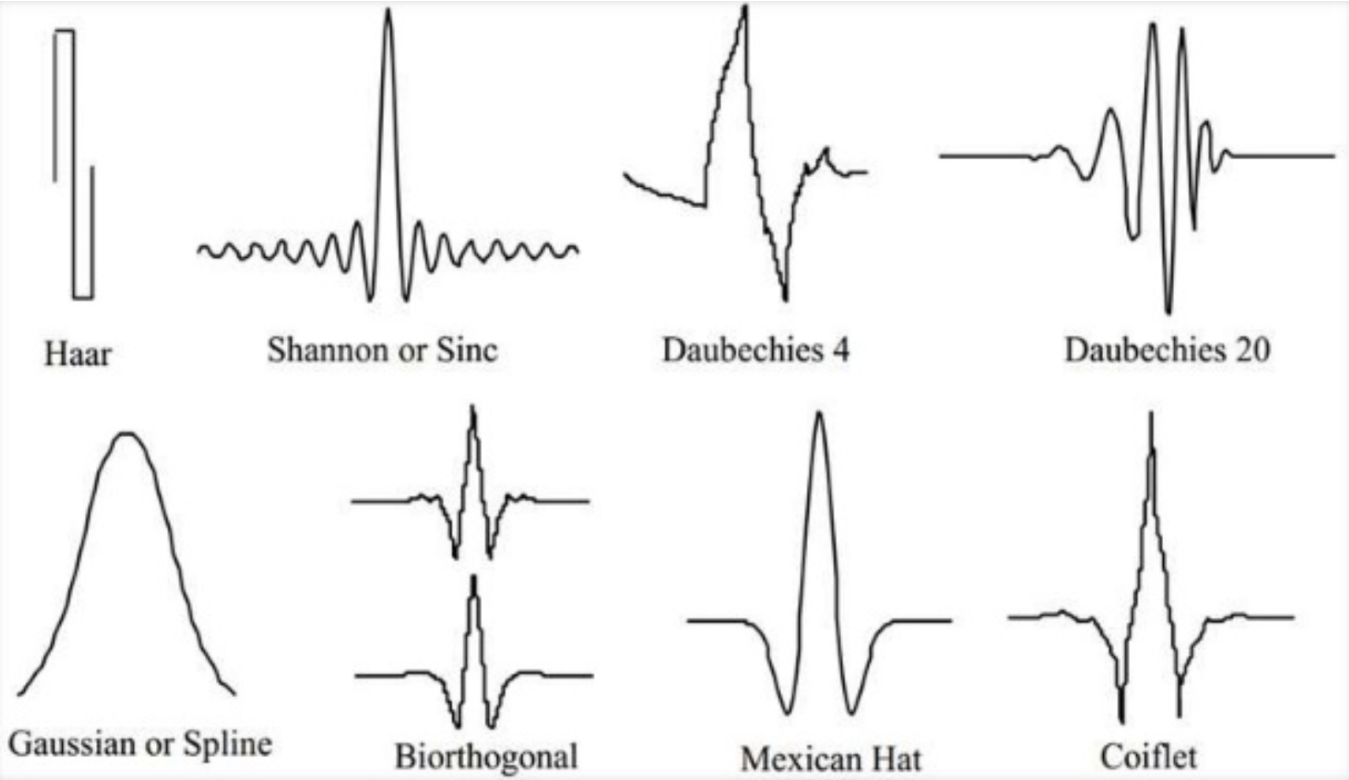
Frequency transformations - Wavelet bases



application depends on:

- the use of the coefficients (filter, store, reconstruct)
- required precision: double, float, half-float, integer, byte
- the need for a *symmertic* base function

Frequency transformations - Wavelet bases



application depends on:

- the use of the coefficients (filter, store, reconstruct)
- required precision: double, float, half-float, integer, byte
- the need for a *symmertic* base function
- the ease of function simplification (i.e. kernelization)

That's it for this week!